Oracle Solaris ZFS-Administrationshandbuch



Teilenr.: 820–2313–15
September 2010

100901@24661

Inhalt

Vorwort

Das *Oracle Solaris ZFS-Administrationshandbuch* enthält Informationen zum Einrichten und Verwalten von Oracle Solaris ZFS-Dateisystemen.

Dieses Handbuch enthält Informationen für SPARC- und x86-basierte Systeme.

Hinweis – Dieses Oracle Solaris-Release unterstützt Systeme, die SPARC- und x86-Prozessorarchitekturen verwenden: UltraSPARC, SPARC64, AMD64, Pentium und Xeon EM64T. Die unterstützten Systeme finden Sie in der *Solaris 10 Hardware-Kompatibilitätsliste* unter http://www.sun.com/bigadmin/hcl. Eventuelle Implementierungsunterschiede zwischen den Plattformtypen sind in diesem Dokument angegeben.

In diesem Handbuch werden x86-Begriffe wie folgt verwendet:

- „x86" bezeichnet die weitere Familie an Produkten, die mit 64-Bit- und 32-Bit-x86-Architekturen kompatibel sind.
- „x64" weist auf spezifische, für 64-Bit-Systeme geltende Informationen zu AMD64- bzw. EM64T-Systemen hin.
- „32-Bit x86" weist auf spezifische, für 32-Bit-Systeme geltende Informationen zu x86-basierten Systemen hin.

Informationen zu unterstützten Systemen finden Sie in der *Solaris 10 Hardware Compatibility List*.

Zielgruppe dieses Handbuchs

Dieses Handbuch richtet sich an alle Personen, die daran interessiert sind, Oracle Solaris ZFS-Dateisysteme einzurichten und zu verwalten. Erfahrungen in der Verwendung des Oracle Solaris Betriebssystems oder einer weiteren UNIX-Version sollten vorhanden sein.

Aufbau dieses Handbuchs

Die folgende Tabelle erläutert die Kapitel dieses Handbuchs.

Kapitel	Beschreibung
Kapitel 1, „Oracle Solaris ZFS-Dateisystem (Einführung)“	Enthält einen Überblick über ZFS und seine Funktionen bzw. Vorteile. Hier werden auch einige Grundkonzepte sowie Begriffe behandelt.
Kapitel 2, „Erste Schritte mit Oracle Solaris ZFS“	Enthält eine schrittweise Anleitung zum Einrichten einfacher ZFS-Konfigurationen mit einfachen Pools und Dateisystemen. Dieses Kapitel enthält darüber hinaus Informationen zu Hardware und Software, die zum Erstellen von ZFS-Dateisysteme erforderlich ist.
Kapitel 3, „Unterschiede zwischen Oracle Solaris ZFS und herkömmlichen Dateisystemen“	Erläutert wichtige Funktionen, durch die sich ZFS erheblich von herkömmlichen Dateisystemen unterscheidet. Durch das Verständnis dieser wichtigen Unterschiede werden Unklarheiten des Zusammenwirkens von herkömmlichen Tools mit ZFS ausgeräumt.
Kapitel 4, „Verwalten von Oracle Solaris ZFS-Speicher-Pools“	Enthält eine ausführliche Beschreibung zur Erstellung und Verwaltung von ZFS-Speicher-Pools.
Kapitel 5, „Installieren und Booten eines Oracle Solaris ZFS-Root-Dateisystems“	Stellt dar, wie ein ZFS-Dateisystem installiert und gebootet wird. Darüber hinaus wird die Migration eines ZFS-Root-Dateisystems in ein ZFS-Dateisystem mithilfe von Oracle Solaris Live Upgrade behandelt.
Kapitel 6, „Verwalten von Oracle Solaris ZFS-Dateisystemen“	Enthält ausführliche Informationen zum Verwalten von ZFS-Dateisystemen. Hier werden Konzepte wie die hierarchische Dateisystemstrukturierung, Eigenschaftsvererbung, die automatische Verwaltung von Einhängepunkten sowie die Interaktion zwischen Netzwerkdateisystemen behandelt.
Kapitel 7, „Arbeiten mit Oracle Solaris ZFS-Snapshots und -Klonen“	Enthält Informationen zum Erstellen und Verwalten von ZFS-Snapshots und -Klonen.
Kapitel 8, „Schützen von Oracle Solaris ZFS-Dateien mit Zugriffssteuerungslisten“	Enthält Informationen zum Arbeiten mit Zugriffssteuerungslisten. Solche Listen schützen ZFS-Dateien, indem im Vergleich zu den UNIX-Standardzugriffsrechten feiner abgestimmte Zugriffsrechte verwendet werden.
Kapitel 9, „Delegierte ZFS-Administration“	Erläutert, wie Benutzern ohne ausreichende Zugriffsrechte mithilfe der delegierten ZFS-Administration die Durchführung von ZFS-Administrationsaufgaben ermöglicht werden kann.
Kapitel 10, „Fortgeschrittene Oracle Solaris ZFS-Themen“	Enthält Informationen zur Verwendung von ZFS-Volumes, von ZFS auf Oracle Solaris-Systemen mit installierten Zonen sowie alternativen Root-Pools.

Kapitel	Beschreibung
Kapitel 11, „Problembehebung und Pool-Wiederherstellung in Oracle Solaris ZFS“	Enthält Informationen zum Erkennen von ZFS-Fehlern und zur Wiederherstellung des Normalzustands nach Auftreten dieser Fehler. Darüber hinaus werden hier auch Maßnahmen zum Vermeiden von Fehlfunktionen beschrieben.
Anhang A, „Oracle Solaris ZFS-Versionsbeschreibungen“	Beschreibung der verfügbaren ZFS-Versionen, der Funktionen jeder Version und des Solaris-Betriebssystems, das die ZFS-Version und die Funktionen bereitstellt.

Verwandte Dokumentation

Die folgenden Handbücher enthalten Informationen zu allgemeinen Themen der Oracle Solaris-Systemadministration:

- *System Administration Guide: Basic Administration*
- *System Administration Guide: Advanced Administration*
- *System Administration Guide: Devices and File Systems*
- *System Administration Guide: Security Services*

Dokumentation, Support und Schulung

Weitere Ressourcen finden Sie auf der folgenden Website:

- Dokumentation (`http://docs.sun.com`)
- Support (`http://www.oracle.com/us/support/systems/index.html`)
- Schulung (`http://education.oracle.com`) – Klicken Sie in der linken Navigationsleiste auf den Sun-Link.

Ihre Meinung ist gefragt

Oracle ist an Ihrer Meinung und an Ihren Anregungen zur Qualität und zum Nutzen der vorliegenden Dokumentation interessiert. Wenn Sie Fehler finden oder Verbesserungsvorschläge haben, gehen Sie zu `http://docs.sun.com`, und klicken Sie dann auf "Feedback". Geben Sie den Titel und die Teilenummer der Dokumentation und wenn möglich das Kapitel, den Abschnitt und die Seitennummer an. Teilen Sie uns bitte mit, ob Sie eine Antwort wünschen.

Das Oracle-Technologienetzwerk (`http://www.oracle.com/technetwork/index.html`) bietet zahlreiche Ressourcen für Oracle-Software:

- Sie können technische Probleme und Lösungen in den Diskussionsforen (`http://forums.oracle.com`) diskutieren.
- Nutzen Sie praktische schrittweise Lernprogramme mit Oracle By Example (`http://www.oracle.com/technology/obe/start/index.html`).
- Laden Sie den Sample Code herunter (`http://www.oracle.com/technology/sample_code/index.html`).

Typografische Konventionen

In der folgenden Tabelle sind die in diesem Handbuch verwendeten typografischen Konventionen aufgeführt.

TABELLE P–1 Typografische Konventionen

Schriftart	Bedeutung	Beispiel
`AaBbCc123`	Die Namen von Befehlen, Dateien, Verzeichnissen sowie Bildschirmausgabe.	Bearbeiten Sie Ihre `.login`-Datei. Verwenden Sie `ls -a`, um eine Liste aller Dateien zu erhalten. `system% Sie haben eine neue Nachricht.`
`AaBbCc123`	Von Ihnen eingegebene Zeichen (im Gegensatz zu auf dem Bildschirm angezeigten Zeichen)	`Computername%` **`su`** `Passwort:`
aabbcc123	Platzhalter: durch einen tatsächlichen Namen oder Wert zu ersetzen	Der Befehl zum Entfernen einer Datei lautet `rm` *Dateiname* .
AaBbCc123	Buchtitel, neue Ausdrücke; hervorgehobene Begriffe	Lesen Sie hierzu Kapitel 6 im *Benutzerhandbuch*. Ein *Cache* ist eine lokal gespeicherte Kopie. Diese Datei *nicht* speichern. **Hinweis:** Einige hervorgehobene Begriffe werden online fett dargestellt.

Shell-Eingabeaufforderungen in Befehlsbeispielen

Die folgende Tabelle zeigt die UNIX-Standardeingabeaufforderung und die Superuser-Eingabeaufforderung für Shells, die im Oracle Solaris-Betriebssystem enthalten sind. Beachten Sie, dass die in den Befehlsbeispielen gezeigte Standardeingabeaufforderung unterschiedlich sein kann, was von der Oracle Solaris-Version abhängt.

TABELLE P–2 Shell-Eingabeaufforderungen

Shell	Eingabeaufforderung
Bash-Shell, Korn-Shell und Bourne-Shell	`$`
Bash-Shell, Korn-Shell und Bourne-Shell für Superuser	`#`
C-Shell	`system%`
C-Shell für Superuser	`system#`

KAPITEL 1

Oracle Solaris ZFS-Dateisystem (Einführung)

Dieses Kapitel bietet einen Überblick über das Oracle Solaris ZFS-Dateisystem sowie seine Funktionen und Vorteile. Darüber hinaus werden die in diesem Handbuch verwendeten Grundbegriffe erläutert.

Dieses Kapitel enthält die folgenden Abschnitte:

Neuerungen in ZFS

In diesem Abschnitt sind die neuen Leistungsmerkmale des ZFS-Dateisystems zusammengefasst.

Teilung eines ZFS-Speicher-Pools mit Datenspiegelung (`zpool split`)

Oracle Solaris 10 9/10: In dieser Solaris-Version können Sie den Befehl `zpool split` verwenden, um einen gespiegelten Speicher-Pool zu teilen, wodurch dem gespiegelten Pool eine oder mehrere Festplatten entnommen werden, um einen weiteren identischen Pool anzulegen.

Weitere Informationen finden Sie unter „Erstellen eines neuen Pools durch Teilen eines ZFS-Speicher-Pools mit Datenspiegelung“ auf Seite 89.

Neuer ZFS-Systemprozess

Oracle Solaris 10 9/10: In dieser Solaris-Version ist jedem ZFS-Speicher-Pool ein Prozess (`zpool-`*poolname*) zugeordnet. Die Teilprozesse dieses Prozesses dienen zur Verarbeitung von E/A-Vorgängen, die den Pool betreffen, wie beispielsweise Komprimierung und Prüfsummenbildung. Dieser Prozess soll Aufschluss über die CPU-Auslastung der einzelnen Speicher-Pools geben. Informationen über diesen Prozess können mithilfe der Befehle `ps` und `prstat` angezeigt werden. Diese Prozesse stehen nur in der globalen Zone zur Verfügung. Weitere Informationen finden Sie unter `SDC(7)`.

Änderungen am Befehl `zpool list`

Oracle Solaris 10 9/10: In dieser Solaris-Version wurde die Ausgabe des Befehls `zpool list` geändert, um bessere Informationen zur Speicherzuordnung bereitzustellen. Beispiel:

```
# zpool list tank
NAME    SIZE  ALLOC   FREE    CAP  HEALTH  ALTROOT
tank    136G  55.2G  80.8G    40%  ONLINE  -
```

Die Felder `USED` und `AVAIL` wurden durch `ALLOC` und `FREE` ersetzt.

Das Feld `ALLOC` ermittelt den physischen Speicherplatz, der allen Datasets und internen Metadaten zugeordnet ist. Das Feld `FREE` ermittelt den im Pool vorhandenen nicht zugeordneten Speicherplatz.

Weitere Informationen finden Sie unter „Anzeigen von Informationen zu ZFS-Speicher-Pools“ auf Seite 106.

Wiederherstellung des ZFS-Speicher-Pools

Oracle Solaris 10 9/10: Ein Speicher-Pool kann beschädigt werden, wenn die betreffenden Geräte nicht verfügbar sind, ein Stromausfall auftritt oder in einer redundanten ZFS-Konfiguration mehr Geräte ausfallen als unterstützt wird. Diese Version bietet neue Funktionen zur Wiederherstellung eines beschädigten Speicher-Pools. Wenn diese Wiederherstellungsfunktionen verwendet werden, können jedoch die letzten Transaktionen, die vor dem Ausfall des Pools stattgefunden haben, verloren gehen.

Zur Wiederherstellung eines beschädigten Pools können die Befehle `zpool clear` und `zpool import` verwendet werden. Diese Befehle unterstützen die Option `-F`. Wenn einer der Befehle `zpool status`, `zpool clear` oder `zpool import` ausgeführt wird, wird die Beschädigung eines Pools automatisch gemeldet. Diese Befehle beschreiben, wie der Pool wiederhergestellt wird.

Weitere Informationen finden Sie unter „Reparieren von Schäden am gesamten ZFS-Speicher-Pool“ auf Seite 324.

Verbesserungen von ZFS-Protokolliergeräten

Oracle Solaris 10 9/10: Folgende Verbesserungen von Protokolliergeräten stehen zur Verfügung:

- Die Eigenschaft `logbias` – Sie können diese Eigenschaft verwenden, um eine Meldung bereitzustellen, die an ZFS übermittelt wird und die synchrone Verarbeitung von Anforderungen für ein bestimmtes Dataset betrifft. Wenn `logbias` auf `latency` gesetzt wird, verwendet ZFS die separaten Protokolliergeräte des Pools (sofern solche Geräte vorhanden sind), um die Anforderungen mit geringer Latenz zu verarbeiten. Wenn `logbias` auf `throughput` gesetzt wird, verwendet ZFS die separaten Protokolliergeräte des Pools nicht. Stattdessen optimiert ZFS die synchronen Vorgänge für den allgemeinen Pool-Durchsatz und die effiziente Nutzung von Ressourcen. Der Standardwert ist `latency`. Der Standardwert wird für die meisten Konfigurationen empfohlen. Durch die Verwendung des Werts `logbias=throughput` kann die Leistung zum Schreiben von Datenbankdateien verbessert werden.
- Entfernen von Protokolliergeräten – Sie können jetzt ein Protokolliergerät aus einem ZFS-Speicher-Pool entfernen, indem Sie den Befehl `zpool remove` verwenden. Ein einzelnes Protokolliergerät kann entfernt werden, indem der Name des Geräts angegeben wird. Ein gespiegeltes Protokolliergerät kann entfernt werden, indem der Spiegel der obersten Hierarchieebene für das Protokoll angegeben wird. Wenn ein separates Protokolliergerät aus dem System entfernt wird, werden ZIL-Transaktionsdaten in den Haupt-Pool geschrieben.

  Redundante virtuelle Geräte der obersten Hierarchieebene werden jetzt durch einen nummerischen Bezeichner identifiziert. In einem gespiegelten Speicher-Pool mit zwei Festplatten beispielsweise ist `mirror-0` das virtuelle Gerät der obersten Hierarchieebene.

  Weitere Informationen finden Sie unter Beispiel 4–3.

RAIDZ-Konfiguration mit dreifacher Parität (`raidz3`)

Release Solaris 10 9/10: In dieser Solaris-Version kann eine redundante RAID-Z-Konfigurationen jetzt einfache, doppelte oder dreifache Parität besitzen. Das bedeutet, dass in einem System ein, zwei oder drei Geräteausfälle ohne Datenverlust möglich sind. Eine RAID-Z-Konfiguration mit doppelter Parität kann mithilfe des Schlüsselworts raidz3 angegeben werden. Weitere Informationen finden Sie unter „Erstellen eines RAID-Z-Speicher-Pools“ auf Seite 74.

Aufbewahren von ZFS-Snapshots

Oracle Solaris 10 9/10: Wenn Sie verschiedene automatische Snapshot-Richtlinien implementieren und dadurch ältere Snapshots durch `zfs receive` gelöscht werden, weil sie nicht mehr auf der Sendeseite vorhanden sind, können Sie die Snapshot-Aufbewahrungsfunktion dieser Solaris-Version verwenden.

Durch die Aufbewahrung eines Snapshots wird verhindert, dass er gelöscht wird. Außerdem ermöglicht diese Funktion das Löschen eines Snapshots zusammen mit Klonen in Abhängigkeit von der Entfernung des letzten Klons mithilfe des Befehls `zfs destroy -d`.

Sie können einen Snapshot oder eine Gruppe von Snapshots aufbewahren. Anhand der folgenden Syntax wird beispielsweise ein Aufbewahrungs-Tag, `keep`, für `tank/home/cindys/snap@1` gesetzt.

```
# zfs hold keep tank/home/cindys@snap1
```

Weitere Informationen finden Sie unter „Aufbewahren von ZFS-Snapshots" auf Seite 227.

Verbesserungen für den Austausch von ZFS-Speichergeräten

Oracle Solaris 10 9/10: In dieser Solaris-Version wird ein Systemereignis (*sysevent*) bereitgestellt, wenn ein Gerät erweitert wird. ZFS wurde verbessert, um diese Ereignisse zu erkennen, und passt den Pool basierend auf der neuen Größe der LU-Nummer an, wobei die Einstellung der Eigenschaft `autoexpand` berücksichtigt wird. Sie können die Pool-Eigenschaft `autoexpand` verwenden, um die automatische Pool-Erweiterung zu aktivieren oder zu deaktivieren, wenn ein dynamisches LU-Nummer-Erweiterungsereignis empfangen wird.

Mithilfe dieser Funktion können Sie eine LU-Nummer erweitern, wodurch der Pool auf den erweiterten Bereich zugreifen kann, ohne einen Pool zu exportieren und zu importieren oder das System neu zu starten.

Die automatische Erweiterung der LU-Nummer ist beispielsweise für den Pool `tank` aktiviert.

```
# zpool set autoexpand=on tank
```

Sie können auch einen Pool erstellen, dessen Eigenschaft `autoexpand` aktiviert ist.

```
# zpool create -o autoexpand=on tank c1t13d0
```

Die Eigenschaft `autoexpand` ist standardmäßig deaktiviert. Sie können entscheiden, ob die LU-Nummer erweitert werden soll oder nicht.

Eine LU-Nummer kann außerdem mithilfe des Befehls `zpool online - e` erweitert werden. Beispiel:

```
# zpool online -e tank c1t6d0
```

Außerdem können Sie, nachdem eine LU-Nummer eingebunden oder verfügbar gemacht wurde, die Eigenschaft `autoexpand` zurücksetzen, indem Sie die Funktion `zpool replace` verwenden. Der folgende Pool wird beispielsweise mit einer 8-GB-Festplatte (`c0t0d0`) erstellt. Die 8-GB-Festplatte wird durch eine 16-GB-Festplatte (`c1t13d0`) ersetzt, aber die Pool-Kapazität wird erst dann erweitert, wenn die Eigenschaft `autoexpand` aktiviert ist.

```
# zpool create pool c0t0d0
# zpool list
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
pool   8.44G  76.5K  8.44G     0%  ONLINE  -
# zpool replace pool c0t0d0 c1t13d0
# zpool list
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
pool   8.44G  91.5K  8.44G     0%  ONLINE  -
# zpool set autoexpand=on pool
# zpool list
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
pool   16.8G   91.5K  16.8G    0%  ONLINE  -
```

Eine weitere Möglichkeit im obigen Beispiel, die LU-Nummer zu erweitern ohne die Eigenschaft `autoexpand` zu aktivieren, ist die Verwendung des Befehls `zpool online - e`. Der Befehl kann auch dann ausgeführt werden, wenn das Gerät bereits in Betrieb genommen ist. Beispiel:

```
# zpool create tank c0t0d0
# zpool list tank
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
tank   8.44G  76.5K  8.44G     0%  ONLINE  -
# zpool replace tank c0t0d0 c1t13d0
# zpool list tank
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
tank   8.44G  91.5K  8.44G     0%  ONLINE  -
# zpool online -e tank c1t13d0
# zpool list tank
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
tank   16.8G    90K  16.8G     0%  ONLINE  -
```

Weitere Verbesserungen für den Austausch von Speichergeräten in dieser Version sind folgende:

- In früheren Versionen war ZFS nicht fähig, eine vorhandene Festplatte durch eine andere Festplatte zu ersetzen oder eine Festplatte einzubinden, wenn die Ersatzfestplatte eine geringfügig unterschiedliche Kapazität hatte. In dieser Version können Sie eine vorhandene Festplatte durch eine andere Festplatte ersetzen oder eine neue Festplatte mit derselben Kapazität einbinden, unter der Voraussetzung, dass der Pool nicht bereits voll ist.
- In dieser Version müssen Sie das System nicht neu starten oder einen Pool exportieren und importieren, um eine LU-Nummer zu erweitern. Wie oben beschrieben können Sie die Eigenschaft `autoexpand` aktivieren oder den Befehl `zpool online - e` verwenden, um eine LU-Nummer auf die volle Kapazität zu erweitern.

Weitere Informationen zum Austauschen von Geräten finden Sie unter „Austauschen von Geräten in einem Speicher-Pool" auf Seite 95.

Unterstützung für ZFS- und Flash-Installation

Solaris 10 10/09: In dieser Solaris-Version haben Sie die Möglichkeit, ein JumpStart-Profil zur Identifizierung eines Flash-Archivs eines ZFS-Root-Pools einzurichten. Weitere Informationen finden Sie unter „Installieren eines ZFS-Root-Dateisystems (Oracle Solaris Flash-Archiv-Installation)" auf Seite 137.

ZFS-Benutzer- und Gruppenkontingente

Solaris 10 10/09: In früheren Solaris-Versionen konnten Kontingente und Reservierungen auf ZFS-Dateisysteme angewendet werden, um Festplattenkapazität zu verwalten und zu reservieren.

In dieser Solaris-Version können Sie ein Kontingent für die Festplattenkapazität einrichten, die von den Dateien eines bestimmten Benutzers oder einer bestimmten Gruppe belegt wird. Das Einrichten von Benutzer- oder Gruppenkontingenten ist in einer Umgebung mit vielen Benutzern oder Gruppen sinnvoll.

Sie können eine Benutzerkontingent mithilfe der Eigenschaft `zfs userquota` einrichten. Um eine Gruppenkontingent einzurichten, verwenden Sie die Eigenschaft `zfs groupquota`. Beispiel:

```
# zfs set userquota@user1=5G tank/data
# zfs set groupquota@staff=10G tank/staff/admins
```

Sie können die aktuelle Einstellung eines Benutzer- oder Gruppenkontingents wie folgt anzeigen:

```
# zfs get userquota@user1 tank/data
NAME       PROPERTY         VALUE            SOURCE
tank/data  userquota@user1  5G               local
# zfs get groupquota@staff tank/staff/admins
NAME               PROPERTY          VALUE             SOURCE
tank/staff/admins  groupquota@staff  10G               local
```

Stellen Sie allgemeine Informationen über ein Kontingent wie folgt ein:

```
# zfs userspace tank/data
TYPE        NAME   USED  QUOTA
POSIX User  root     3K   none
POSIX User  user1     0    5G

# zfs groupspace tank/staff/admins
TYPE         NAME   USED  QUOTA
```

```
POSIX Group  root      3K   none
POSIX Group  staff      0    10G
```

Sie können die von einem einzelnen Benutzer belegte Festplattenkapazität mithilfe der Eigenschaften `userused@`*user* anzeigen. Die von einer Gruppe belegte Festplattenkapazität kann mithilfe der Eigenschaft `groupused@` *group* angezeigt werden. Beispiel:

```
# zfs get userused@user1 tank/staff
NAME        PROPERTY        VALUE           SOURCE
tank/staff  userused@user1  213M            local
# zfs get groupused@staff tank/staff
NAME        PROPERTY         VALUE            SOURCE
tank/staff  groupused@staff  213M             local
```

Weitere Informationen zum Einrichten von Benutzerkontingenten finden Sie unter „Einstellen von ZFS-Kontingenten und -Reservierungen" auf Seite 218.

ZFS-Zugriffssteuerungslistenvererbungsmodus "Pass Through" zur Ausführungsberechtigung

Solaris 10 10/09: In früheren Solaris-Versionen konnte die Zugriffssteuerungslistenvererbung angewendet werden, sodass alle Dateien mit den Zugriffsrechten `0664` oder `0666` erstellt wurden. Wenn Sie die Ausführungsberechtigung aus dem Dateierstellungsmodus optional in die vererbte Zugriffssteuerungsliste einschließen möchten, können Sie in dieser Version den Modus `aclinherit` verwenden, um die Ausführungsberechtigung auf die vererbte Zugriffssteuerungsliste zu übertragen.

Wenn `aclinherit=passthrough-x` für ein ZFS-Dataset aktiviert ist, können Sie die Ausführungsberechtigung für eine Ausgabedatei einschließen, die mit dem Compiler-Tool `cc` oder `gcc` erstellt wurde. Beinhaltet die vererbbare Zugriffssteuerungsliste die Ausführungsberechtigung nicht, so ist die Ausgabe des Compilers erst dann ausführbar, wenn Sie mit dem Befehl `chmod` die Berechtigungen der Datei ändern.

Weitere Informationen finden Sie unter Beispiel 8–12.

Verbesserungen der ZFS-Eigenschaften

Solaris 10 10/09 und Oracle Solaris 10 9/10: Diese Versionen enthalten folgende Verbesserungen des ZFS-Dateisystems.

- **Verbesserung der Eigenschaften von ZFS-Snapshot-Datenströmen** – Sie können eine empfangene Eigenschaft setzen, die sich von ihrer lokalen Einstellung unterscheidet. Sie empfangen beispielsweise einen Datenstrom, dessen Komprimierungseigenschaft deaktiviert ist, möchten aber, dass die Komprimierung im Dateisystem, das die Daten empfängt, aktiviert ist. Der empfangene Datenstrom weist den empfangenen

Komprimierungswert `off` und den lokalen Komprimierungswert `on` auf. Da der lokale Wert den empfangenen Wert übersteuert, müssen Sie sich nicht darum kümmern, dass die Einstellung auf der Sendeseite den Wert auf der Empfangsseite ersetzt. Der Befehl `zfs get` zeigt den effektiven Wert der Komprimierungseigenschaft in der Spalte `VALUE`.

Es folgen neue ZFS-Befehlsoptionen und Eigenschaften, die gesendete und lokale Eigenschaftswerte unterstützen:

- Verwenden Sie `zfs inherit -S`, um einen lokalen Eigenschaftswert auf den empfangenen Wert (sofern vorhanden) zurückzusetzen. Wenn eine Eigenschaft keinen empfangenen Wert aufweist, ist das Verhalten des Befehls `zfs inherit - S` dasselbe wie das des Befehls `zfs inherit` ohne die Option `-S`. Wenn die Eigenschaft einen empfangenen Wert aufweist, maskiert der Befehl `zfs inherit` den empfangenen Wert mit dem vererbten Wert, bis dieser durch Ausgabe des Befehls `zfs inherit -S` auf den empfangenen Wert zurückgesetzt wird.
- Sie können den Befehl `zfs get -o` verwenden, um die neue nicht standardmäßige Spalte `RECEIVED` einzubeziehen. Sie können aber auch den Befehl `zfs get -o all` verwenden, um alle Spalten einschließlich `RECEIVED` einzubeziehen.
- Sie können die Option `zfs send -p` verwenden, um Eigenschaften in zu sendenden Datenstrom ohne die Option `-R` einzubeziehen.

Außerdem können Sie die Option `zfs send -e` verwenden, um mithilfe des letzten Elements des gesendeten Snapshot-Namens den Namen des neuen Snapshots festzulegen. Im folgenden Beispiel wird der Snapshot `poola/bee/cee@1` an das Dateisystem `poold/eee` gesendet, und nur das letzte Element (`cee@1`) des Snapshot-Namens wird verwendet, um das Dateisystem und den Snapshot zu erstellen.

```
# zfs list -rt all poola
NAME               USED  AVAIL  REFER  MOUNTPOINT
poola              134K   134G    23K  /poola
poola/bee           44K   134G    23K  /poola/bee
poola/bee/cee       21K   134G    21K  /poola/bee/cee
poola/bee/cee@1       0      -    21K  -
# zfs send -R poola/bee/cee@1 | zfs receive -e poold/eee
# zfs list -rt all poold
NAME               USED  AVAIL  REFER  MOUNTPOINT
poold              134K   134G    23K  /poold
poold/eee           44K   134G    23K  /poold/eee
poold/eee/cee       21K   134G    21K  /poold/eee/cee
poold/eee/cee@1       0      -    21K  -
```

- **Einstellung der Eigenschaften von ZFS-Dateisystemen zum Zeitpunkt der Erstellung eines Pools** – Sie können die Eigenschaften des ZFS-Dateisystems bei der Erstellung eines Speicher-Pool einstellen. Im folgenden Beispiel ist die Komprimierung auf dem ZFS-Dateisystem, das bei Erstellung des Pools eingerichtet wird, aktiviert:

```
# zpool create -O compression=on pool mirror c0t1d0 c0t2d0
```

- **Einstellung von Cache-Eigenschaften für ein ZFS-Dateisystem** – Zwei neue Eigenschaften von ZFS-Dateisystemen werden zur Verfügung gestellt, mit denen Sie kontrollieren können, was im primären Cache (ARC) und im sekundären Cache (L2ARC) gespeichert wird. Die Cache-Eigenschaften sind folgendermaßen eingestellt:
  - `primarycache` – Kontrolliert, was im ARC gespeichert wird.
  - `secondarycache` – Kontrolliert, was im L2ARC gespeichert wird.
  - Mögliche Werte für beide Eigenschaften – `all`, `none` und `metadata`. Ist diese Eigenschaft auf `all` gesetzt, werden sowohl Benutzerdaten als auch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `none` gesetzt, werden weder Benutzerdaten noch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `metadata` gesetzt, werden nur Metadaten im Cache gespeichert. Die Standardeinstellung ist `all`.

  Sie können diese Eigenschaften für ein bereits vorhandenes Dateisystem oder bei der Erstellung eines Dateisystems einstellen. Beispiel:

  ```
  # zfs set primarycache=metadata tank/datab
  # zfs create -o primarycache=metadata tank/newdatab
  ```

  Sobald diese Eigenschaften für bereits vorhandene Dateisysteme eingestellt werden, werden nur neue E/A-Vorgänge auf der Basis des Werts dieser Eigenschaften im Cache gespeichert.

  Bei einigen Datenbank-Umgebungen kann es von Vorteil sein, Benutzerdaten nicht im Cache zu speichern. Sie müssen selbst bestimmen, ob das Einstellen von Cache-Eigenschaften für Ihre Umgebung sinnvoll ist.
- **Anzeigen von Eigenschaften zur Berechnung von Festplattenkapazität** – Mit neuen schreibgeschützten Dateisystem-Eigenschaften können Sie die Festplattenkapazitätsbelegung für Klone, Dateisysteme, Volumes und Snapshots bestimmen. Folgende Eigenschaften stehen zur Verfügung:
  - `usedbychildren` – Gibt Festplattenkapazität an, die von untergeordneten Objekten dieses Datasets beansprucht wird und die beim Löschen dieser untergeordneten Objekte frei werden würde. Die Abkürzung für die Eigenschaft lautet `usedchild`
  - `usedbydataset` – Gibt Festplattenkapazität an, die vom Dataset selbst beansprucht wird und die beim Löschen des Datasets und vorherigem Löschen aller Snapshots sowie Entfernen aller Reservierungen frei werden würde. Die Abkürzung für die Eigenschaft lautet `usededds`
  - `usedbyrefreservation` – Gibt die von einem `refreservation`-Set dieses Datasets beanspruchte Festplattenkapazität an, die beim Entfernen der Reservierung frei werden würde. Die Abkürzung für die Eigenschaft lautet `usedrefreserv`
  - `usedbysnapshots` – Gibt Festplattenkapazität an, die von Snapshots dieses Datasets beansprucht wird und die beim Löschen dieser Snapshots frei werden würde. Beachten Sie, dass es sich dabei nicht um die Summe der `used`-Eigenschaften der Snapshots handelt, da Festplattenkapazität von mehreren Snapshots gemeinsam genutzt werden kann. Die Abkürzung für die Eigenschaft lautet `usedsnap`

Diese neuen Eigenschaften teilen den Wert der `used` Eigenschaft in die verschiedenen Elemente auf, die Festplattenkapazität beanspruchen. Genauer betrachtet wird der Wert der `used` Eigenschaft folgendermaßen aufgeteilt:

```
used property = usedbychildren + usedbydataset + usedbyrefreservation + usedbysnapshots
```

Sie können diese Eigenschaften mit dem Befehl `zfs list - o space` genauer betrachten. Beispiel:

```
$ zfs list -o space
NAME               AVAIL   USED  USEDSNAP  USEDDS  USEDREFRESERV  USEDCHILD
rpool              25.4G  7.79G         0     64K              0      7.79G
rpool/ROOT         25.4G  6.29G         0     18K              0      6.29G
rpool/ROOT/snv_98  25.4G  6.29G         0   6.29G              0          0
rpool/dump         25.4G  1.00G         0   1.00G              0          0
rpool/export       25.4G    38K         0     20K              0        18K
rpool/export/home  25.4G    18K         0     18K              0          0
rpool/swap         25.8G   512M         0    111M           401M          0
```

Der obige Befehl entspricht dem Befehl `zfs list - o name,avail,used,usedsnap,usedds,usedrefreserv,usedchild -t filesystem,volume`.

- **Snapshots auflisten**– Die Pool-Eigenschaft `listsnapshots` steuert, ob Snapshot-Informationen mit dem Befehl `zfs list` angezeigt werden. Der Standardwert ist `on`, Snapshot-Informationen werden also standardmäßig angezeigt.

  Wenn in Ihrem System viele ZFS-Snapshots vorhanden sind und Sie verhindern möchten, dass Snapshot-Informationen mithilfe des Befehls `zfs list` angezeigt werden, deaktivieren Sie die Eigenschaft `listsnapshots` wie folgt:

```
# zpool get listsnapshots pool
NAME  PROPERTY       VALUE      SOURCE
pool  listsnapshots  on         default
# zpool set listsnaps=off pool
```

  Wenn Sie die Eigenschaft `listsnapshots` deaktivieren, können Sie den Befehl `zfs list -t snapshots` verwenden, um Snapshot-Informationen anzuzeigen. Beispiel:

```
# zfs list -t snapshot
NAME                     USED  AVAIL  REFER  MOUNTPOINT
pool/home@today           16K      -    22K  -
pool/home/user1@today       0      -    18K  -
pool/home/user2@today       0      -    18K  -
pool/home/user3@today       0      -    18K  -
```

Wiederherstellung von ZFS-Protokolliergeräten

Solaris 10 10/09: In dieser Version identifiziert ZFS Intent-Protokoll-Fehler durch den Befehl `zpool status`. Diese Fehler werden auch von der Fault Management Architecture (FMA) gemeldet. Ein Intent-Protokoll-Fehler kann sowohl mit ZFS als auch mit FMA behoben werden.

Wenn das System beispielsweise plötzlich herunterfährt, bevor synchrone Schreibvorgänge in einem Pool mit separatem Protokolliergerät abgeschlossen werden können, werden Meldungen wie die folgenden angezeigt:

```
# zpool status -x
  pool: pool
 state: FAULTED
status: One or more of the intent logs could not be read.
        Waiting for adminstrator intervention to fix the faulted pool.
action: Either restore the affected device(s) and run 'zpool online',
        or ignore the intent log records by running 'zpool clear'.
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        FAULTED      0     0     0 bad intent log
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
        logs        FAULTED      0     0     0 bad intent log
          c0t5d0    UNAVAIL      0     0     0 cannot open
```

Nach einem Ausfall des Protokolliergeräts können Sie den Normalbetrieb folgendermaßen wiederherstellen:

- Ersetzen Sie das Protokolliergerät oder stellen Sie es wieder her. In diesem Beispiel handelt es sich um das Protokolliergerät `c0t5d0`.
- Nehmen Sie das Protokolliergerät wieder in Betrieb.

  ```
  # zpool online pool c0t5d0
  ```
- Setzen Sie den Fehlerzustand "Protokolliergerät fehlgeschlagen" zurück.

  ```
  # zpool clear pool
  ```

Wenn Sie den Normalbetrieb wiederherstellen möchten, ohne das Protokolliergerät zu ersetzen, können Sie den Fehler mit dem Befehl `zpool clear` löschen. In diesem Szenario läuft der Pool in eingeschränktem Modus, und die Protokolleinträge werden in den Haupt-Pool geschrieben, bis das separate Protokolliergerät ersetzt wurde.

Verwenden Sie gespiegelte Protokolliergeräte, um den Ausfall von Protokolliergeräten zu vermeiden.

Verwenden von Cache-Geräten im ZFS-Speicher-Pool

Solaris 10 10/09: In dieser Version können Sie Pools erstellen und *Cache-Geräte* angeben, die zur Speicherung von Speicher-Pool-Daten im Cache dienen.

Cache-Speicher·bietet zwischen Hauptspeicher und Festplatte eine zusätzliche Schicht zur Datenspeicherung. Cache-Geräte bieten die größte Leistungsverbesserung für die Direktspeicherung von Daten mit vorwiegend statischem Inhalt.

Beim Erstellen eines Speicher-Pools können eines oder mehrere Cache-Speichergeräte angegeben werden. Beispiel:

```
# zpool create pool mirror c0t2d0 c0t4d0 cache c0t0d0
# zpool status pool
  pool: pool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
        cache
          c0t0d0    ONLINE       0     0     0

errors: No known data errors
```

Nach dem Erstellen von Cache-Speichergeräten werden diese nach und nach mit Daten aus dem Hauptspeicher gefüllt. Je nach Größe eines definierten Cache-Speichergeräts kann es bis zu über eine Stunde lang dauern, bis das Gerät voll ist. Die Kapazität und Lesevorgänge können mithilfe des Befehls `zpool iostat` wie folgt überwacht werden:

```
# zpool iostat -v pool 5
```

Nach der Erstellung eines Pools können Cache-Geräte dem Pool hinzugefügt oder aus dem Pool entfernt werden.

Weitere Informationen finden Sie unter „Erstellen eines ZFS-Speicher-Pools mit Cache-Geräten“ auf Seite 76 und Beispiel 4–4.

Zonenmigration in einer ZFS-Umgebung

Solaris 10 5/09: Diese Version bietet eine erweiterte Unterstützung für das Migrieren von Zonen in einer ZFS-Umgebung mit Oracle Solaris Live Upgrade. Weitere Informationen finden Sie unter „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (ab Solaris 10 5/09)“ auf Seite 156.

Die Solaris 10 5/09-Versionshinweise enthalten eine Liste der bekannten Probleme mit diesem Release.

Unterstützung für Installation und Booten von ZFS-Root-Dateisystemen

Solaris 10 10/08: Mit dieser Solaris-Version können ZFS-Root-Dateisysteme installiert und gebootet werden. Das Installieren eines ZFS-Root-Dateisystems ist sowohl mit der Erstinstallationsoption als auch mit der JumpStart-Funktion möglich. Sie können aber auch Oracle Solaris Live Upgrade verwenden, um ein UFS-Root-Dateisystem auf ein ZFS-Root-Dateisystem zu migrieren. Außerdem steht ZFS-Unterstützung für Swap- und Dump-Geräte zur Verfügung. Weitere Informationen finden Sie in Kapitel 5, „Installieren und Booten eines Oracle Solaris ZFS-Root-Dateisystems“.

Eine Liste der bekannten Probleme mit dieser Version finden Sie unter:

http://hub.opensolaris.org/bin/view/Community+Group+zfs/boot

Ziehen Sie auch die Versionshinweise für Solaris 10 10/08 heran.

Wiederherstellen eines Datasets ohne Aushängen

Solaris 10 10/08: Diese Version bietet die Möglichkeit, ein Dataset wiederherzustellen, ohne es zuvor auszuhängen. Das bedeutet, dass die Option `zfs rollback -f` zum Erzwingen des Aushängens nicht mehr gebraucht wird. Die Option `-f` wird nicht mehr unterstützt und wird bei Angabe ignoriert.

Verbesserungen des Befehls `zfs send`

Solaris 10 10/08: Diese Version bietet die folgenden Verbesserungen des Befehls `zfs send`. Mit diesem Befehl können Sie folgende Aufgaben ausführen:

- Senden aller inkrementellen Datenströme von einem Snapshot zu einem kumulativen Snapshot. Beispiel:

  ```
  # zfs list
  NAME                      USED  AVAIL  REFER  MOUNTPOINT
  pool                      428K  16.5G    20K  /pool
  pool/fs                    71K  16.5G    21K  /pool/fs
  pool/fs@snapA              16K      -  18.5K  -
  pool/fs@snapB              17K      -    20K  -
  pool/fs@snapC              17K      -  20.5K  -
  pool/fs@snapD                0      -    21K  -
  # zfs send -I pool/fs@snapA pool/fs@snapD > /snaps/fs@combo
  ```

  Mit dieser Syntax werden alle inkrementellen Snapshots zwischen `fs@snapA` und `fs@snapD` nach `fs@combo` gesendet.

- Senden eines inkrementellen Datenstroms vom ursprünglichen Snapshot, um einen Klon zu erstellen. Der ursprüngliche Snapshot muss auf der Empfangsseite bereits vorhanden sein, damit der inkrementelle Datenstrom angenommen werden kann. Beispiel:

```
# zfs send -I pool/fs@snap1 pool/clone@snapA > /snaps/fsclonesnap-I
.
.
# zfs receive -F pool/clone < /snaps/fsclonesnap-I
```

- Senden eines Replikationsstreams aller abhängigen Dateisysteme zu den benannten Snapshots. Nach dem Empfang werden alle Eigenschaften, Snapshots, abhängigen Dateisysteme und Klone beibehalten. Beispiel:

  ```
  # zfs send -R pool/fs@snap > snaps/fs-R
  ```

  Ein ausführlicheres Beispiel finden Sie in Beispiel 7–1.
- Senden eines inkrementellen Replikationsstreams. Beispiel:

  ```
  # zfs send -R -[iI] @snapA pool/fs@snapD
  ```

  Ein ausführlicheres Beispiel finden Sie in Beispiel 7–1.

Weitere Informationen finden Sie unter „Senden und Empfangen komplexer ZFS-Snapshot-Datenströme“ auf Seite 238.

ZFS-Kontingente und -Reservierungen ausschließlich für Dateisystemdaten

Solaris 10 10/08: Zusätzlich zu den vorhandenen ZFS-Funktionen für Kontingente und Reservierungen enthält diese Version auch Dataset-Kontingente und -Reservierungen. Diese Kontingente und Reservierungen berücksichtigen bei der Berechnung von Festplattenkapazität keine untergeordneten Objekte wie z. B. Snapshots oder Klone.

- Die Eigenschaft `refquota` erzwingt einen absoluten Grenzwert der Festplattenkapazität, die von einem Dataset belegt werden kann. Dieser absolute Grenzwert berücksichtigt jedoch nicht die Festplattenkapazität, die von untergeordneten Objekten wie z. B. Snapshots oder Klonen belegt wird.
- Die Eigenschaft `refreservation` legt die für ein Dataset minimal garantierte Festplattenkapazität fest (Festplattenkapazität für untergeordnete Objekte wird nicht berücksichtigt).

Sie können beispielsweise mit `refquota` einen Grenzwert von 10 GB für `studentA` setzen, um einen absoluten Grenzwert von 10 GB für referenzierte Festplattenkapazität festzulegen. Zum Erreichen einer zusätzlichen Flexibilität können Sie ein Kontingent von 20 GB festlegen, mit dessen Hilfe Sie die Snapshots von `studentA` verwalten können.

```
# zfs set refquota=10g tank/studentA
# zfs set quota=20g tank/studentA
```

Weitere Informationen finden Sie unter „Einstellen von ZFS-Kontingenten und -Reservierungen“ auf Seite 218.

Eigenschaften von ZFS-Speicher-Pools

Solaris 10 10/08: Die Eigenschaften von ZFS-Speicher-Pools wurden mit einer früheren Version eingeführt. Diese Version bietet zwei Eigenschaften: `cachefile` und `failmode`.

Im Folgenden werden die neuen Speicher-Pool-Eigenschaften dieser Version beschrieben:

- Die Eigenschaft `cachefile` – Mit dieser Eigenschaft wird bestimmt, wo Pool-Konfigurationsinformationen im Cache gespeichert werden. Alle Pools im Cache werden beim Booten des Systems automatisch importiert. Installations- und Cluster-Umgebungen können jedoch erfordern, dass diese Informationen an anderer Stelle im Cache gespeichert werden, sodass Pools nicht automatisch importiert werden.

  Sie können diese Eigenschaft so einstellen, dass Poolkonfigurationen an einer anderen Stelle im Cache-Speicher abgelegt werden und später mithilfe des Befehls `zpool import -c` importiert werden können. Für die meisten ZFS-Konfigurationen wird diese Eigenschaft nicht verwendet.

  Die Eigenschaft `cachefile` ist nicht beständig und wird nicht auf Festplatte gespeichert. Diese Eigenschaft löst die Eigenschaft `temporary` ab, die in früheren Solaris-Versionen anzeigte, dass Poolinformationen nicht im Cache gespeichert werden sollten.
- Die Eigenschaft `failmode` – Mit dieser Eigenschaft wird festgelegt, wie sich das System im Falle eines äußerst schwerwiegenden Poolausfalls verhalten soll, der auf Verlust der Gerätekonnektivität oder den gleichzeitigen Ausfall aller Speichergeräte im Pool zurückzuführen ist. Die Eigenschaft `failmode` kann auf die Werte `wait`, `continue` oder `panic` gesetzt werden. Der Standardwert ist `wait`. Das bedeutet, dass Sie das ausgefallene Gerät neu in den Pool integrieren oder auswechseln und den Fehler danach mit dem Befehl `zpool clear` löschen müssen.

  Die Eigenschaft `failmode` wird wie andere einstellbare ZFS-Eigenschaften auch gesetzt. Dies kann vor oder nach dem Erstellen eines Pools geschehen. Beispiel:

  ```
  # zpool set failmode=continue tank
  # zpool get failmode tank
  NAME  PROPERTY  VALUE     SOURCE
  tank  failmode  continue  local
  ```

  ```
  # zpool create -o failmode=continue users mirror c0t1d0 c1t1d0
  ```

  Eine Beschreibung dieser Eigenschaften können Sie Tabelle 4–1 entnehmen.

Verbesserungen des ZFS-Befehlsprotokolls (`zpool history`)

Solaris 10 10/08: Der Befehl `zpool history` wurde um die folgenden neuen Leistungsmerkmale erweitert:

- Es werden jetzt Informationen zu Ereignissen im ZFS-Dateisystem angezeigt. Beispiel:

```
# zpool history
History for 'rpool':
2010-06-23.09:30:12 zpool create -f -o failmode=continue -R /a -m legacy -o
cachefile=/tmp/root/etc/zfs/zpool.cache rpool c1t0d0s0
2010-06-23.09:30:13 zfs set canmount=noauto rpool
2010-06-23.09:30:13 zfs set mountpoint=/rpool rpool
2010-06-23.09:30:13 zfs create -o mountpoint=legacy rpool/ROOT
2010-06-23.09:30:14 zfs create -b 8192 -V 2048m rpool/swap
2010-06-23.09:30:14 zfs create -b 131072 -V 1024m rpool/dump
2010-06-23.09:30:15 zfs create -o canmount=noauto rpool/ROOT/zfsBE
2010-06-23.09:30:16 zpool set bootfs=rpool/ROOT/zfsBE rpool
2010-06-23.09:30:16 zfs set mountpoint=/ rpool/ROOT/zfsBE
2010-06-23.09:30:16 zfs set canmount=on rpool
2010-06-23.09:30:16 zfs create -o mountpoint=/export rpool/export
2010-06-23.09:30:17 zfs create rpool/export/home
```

- Die Option `-l` kann verwendet werden, um ein langes Format anzuzeigen. Dieses Format enthält den Benutzernamen, den Hostnamen und die Zone, in der der Vorgang ausgeführt wurde. Beispiel:

```
# zpool history -l rpool
History for 'tank':
2010-06-24.13:07:58 zpool create tank mirror c2t2d0 c2t5d0 [user root on neo:global]
2010-06-24.13:08:23 zpool scrub tank [user root on neo:global]
2010-06-24.13:38:42 zpool clear tank [user root on neo:global]
2010-06-29.11:44:18 zfs create tank/home [user root on neo:global]
2010-06-29.13:28:51 zpool clear tank c2t5d0 [user root on neo:global]
2010-06-30.14:07:40 zpool add tank spare c2t1d0 [user root on neo:global
```

- Die Option `-i` kann zum Anzeigen interner Ereignisinformationen verwendet werden, die bei der Diagnose behilflich sein können. Beispiel:

```
# zpool history -i tank
History for 'tank':
2010-06-24.13:07:58 zpool create tank mirror c2t2d0 c2t5d0
2010-06-24.13:08:23 [internal pool scrub txg:6] func=1 mintxg=0 maxtxg=6
2010-06-24.13:08:23 [internal pool create txg:6] pool spa 22; zfs spa 22; zpl 4; uts neo 5.10 Generic_142909-13 s
2010-06-24.13:08:23 [internal pool scrub done txg:6] complete=1
2010-06-24.13:08:23 zpool scrub tank
2010-06-24.13:38:42 zpool clear tank
2010-06-24.13:38:42 [internal pool scrub txg:69] func=1 mintxg=3 maxtxg=8
2010-06-24.13:38:42 [internal pool scrub done txg:69] complete=1
2010-06-29.11:44:18 [internal create txg:14241] dataset = 34
2010-06-29.11:44:18 zfs create tank/home
2010-06-29.13:28:51 zpool clear tank c2t5d0
2010-06-30.14:07:40 zpool add tank spare c2t1d0
```

Weitere Informationen zur Verwendung des Befehls `zpool history` entnehmen Sie bitte dem Abschnitt „Beheben von Problemen mit ZFS" auf Seite 303.

Upgrade von ZFS-Dateisystemen (zfs upgrade)

Solaris 10 10/08: Diese Version bietet den Befehl `zfs upgrade`, mit dem vorhandene Dateisysteme aktualisiert werden können, um neue Verbesserungen des ZFS-Dateisystems zu nutzen. ZFS-Speicher-Pools besitzen eine ähnliche Upgrade-Funktion, um vorhandene Speicher-Pools um neue Funktionalität zu erweitern.

Beispiel:

```
# zfs upgrade
This system is currently running ZFS filesystem version 3.

All filesystems are formatted with the current version.
```

Hinweis – Auf aktualisierte Dateisysteme und Datenströme, die aus diesen aktualisierten Dateisystemen mithilfe des Befehls `zfs send` erstellt wurden, kann nicht auf Systemen zugegriffen werden, auf denen ältere Softwareversion laufen.

Delegierte ZFS-Administration

Solaris 10 10/08: In dieser Version können Benutzern ohne Zugriffsrechte genau definierte Zugriffsrechte für die Durchführung von ZFS-Administrationsaufgaben gewährt werden.

Zum Gewähren und Verweigern von Zugriffsrechten dienen die Befehle `zfs allow` und `zfs unallow`.

Mit der Pool-Eigenschaft `delegation` kann die delegierte Administration modifiziert werden. Beispiel:

```
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  on          default
# zpool set delegation=off users
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  off         local
```

Standardmäßig ist die Eigenschaft `delegation` aktiviert.

Weitere Informationen entnehmen Sie bitte Kapitel 9, „Delegierte ZFS-Administration" und `zfs`(1M).

Einrichten separater ZFS-Protokolliergeräte

Solaris 10 10/08: Das Protokoll ZIL (ZFS Intent Log) erfüllt die POSIX-Anforderungen für synchrone Transaktionen. So setzen Datenbanken bei der Rückkehr von Systemaufrufen beispielsweise oft voraus, dass Transaktionen auf stabilen Speichergeräten stattfinden. NFS und andere Anwendungen können zur Gewährleistung der Datenstabilität ebenfalls `fsync()` verwenden. Standardmäßig wird das ZIL aus Blöcken innerhalb des Hauptspeicherpools zugewiesen. In dieser Solaris-Version können Sie festlegen, ob ZIL-Blöcke weiterhin vom Hauptspeicherpool oder von einem separaten Protokolliergerät zugewiesen werden sollen. Durch Verwendung separater Intent-Protokolliergeräte im ZFS-Speicher-Pool wie z. B. NVRAM oder eine spezielle Festplatte kann jedoch eine höhere Leistung erzielt werden.

Protokolliergeräte für das Intent-Protokoll von ZFS sind etwas Anderes als Datenbankprotokolldateien.

Sie können ZFS-Protokolliergeräte während oder nach der Erstellung eines Speicher-Pools einrichten. Beispiele zum Einrichten von Protokolliergeräten finden Sie unter „Erstellen eines ZFS-Speicher-Pools mit Protokolliergeräten“ auf Seite 75 und „Hinzufügen von Datenspeichergeräten zu einem Speicher-Pool“ auf Seite 83.

Sie können zur Datenspiegelung an ein vorhandenes Protokolliergerät ein weiteres Protokolliergerät anschließen. Dies entspricht dem Verbinden eines Speichergeräts in einem Speicher-Pool ohne Datenspiegelung.

Berücksichtigen Sie folgende Aspekte bei der Überlegung, ob die Einrichtung eines ZFS-Protokolliergeräts für Ihre Umgebung ratsam ist:

- Jegliche Leistungssteigerung durch die Implementierung eines separaten Protokolliergeräts ist von der Art des Geräts, der Hardwarekonfiguration des Speicher-Pools sowie von der Arbeitslast der Anwendung abhängig. Vorläufige Informationen zur Leistung finden Sie in diesem Blog:

  `http://blogs.sun.com/perrin/entry/slog_blog_or_blogging_on`
- Es ist möglich, Protokolliergeräte zu spiegeln oder deren Replikation aufzuheben. RAID-Z wird für Protokolliergeräte jedoch nicht unterstützt.
- Wenn ein separates Protokolliergerät nicht gespiegelt wurde und das Gerät mit den Protokollen ausfällt, wird durch Speicherung von Protokollblöcken auf den Speicher-Pool zurückgeschaltet.
- Protokolliergeräte können als Teil des Speicher-Pools hinzugefügt, ersetzt, angehängt, ausgehängt, importiert und exportiert werden. Protokolliergeräte können ab Solaris-Version 10 9/10 entfernt werden.
- Die Mindestgröße für ein Protokolliergerät entspricht der Mindestkapazität für jedes Gerät in einem Pool. Dies sind 64 MB. Auf einem Protokolliergerät können verhältnismäßig wenige Ausführungsdaten gespeichert werden. Bei Bestätigung der Protokolltransaktion (Systemaufruf) werden die Protokollblöcke geleert.
- Ein Protokolliergerät sollte nicht größer als rund die Hälfte des physischen Speichers sein, da dies die Höchstmenge an potenziellen Ausführungsdaten ist, die gespeichert werden kann. So sollten Sie beispielsweise für ein System mit 16 GB physischem Speicher ein Protokolliergerät mit 8 GB Speicher in Erwägung ziehen.

Erstellen intermediärer ZFS-Datasets

Solaris 10 10/08: Durch Verwendung der Option `-p` mit den Befehlen `zfs create`, `zfs clone` und `zfs rename` können Sie schnell ein intermediäres Dataset erstellen, falls es noch nicht vorhanden ist.

Das folgende Beispiel zeigt, wie ZFS-Datasets (`users/area51`) im Speicher-Pool `datab` erstellt werden.

```
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
datab                       106K  16.5G    18K  /datab
# zfs create -p -o compression=on datab/users/area51
```

Wenn während des Erstellungsvorgangs bereits ein intermediäres Dataset vorhanden ist, wird er ohne Fehlermeldung abgeschlossen.

Angegebene Eigenschaften gelten für das Ziel-Dataset und nicht für die intermediären Datasets. Beispiel:

```
# zfs get mountpoint,compression datab/users/area51
NAME                PROPERTY     VALUE                SOURCE
datab/users/area51  mountpoint   /datab/users/area51  default
datab/users/area51  compression  on                   local
```

Es wird ein intermediäres Dataset mit Standard-Einhängepunkt erstellt. Alle zusätzlichen Eigenschaften werden für dieses intermediäre Dataset deaktiviert. Beispiel:

```
# zfs get mountpoint,compression datab/users
NAME         PROPERTY     VALUE         SOURCE
datab/users  mountpoint   /datab/users  default
datab/users  compression  off           default
```

Weitere Informationen finden Sie in der Manpage `zfs`(1M).

Verbesserungen für den Austausch von ZFS-Speichergeräten bei laufendem Betrieb

Solaris 10 10/08: Mit dieser Version reagiert ZFS effektiver auf Speichergeräte, die entfernt werden. Außerdem können eingefügte Speichergeräte jetzt automatisch erkannt werden.

- Sie können ein Speichergerät durch ein anderes auswechseln, ohne dafür den Befehl `zpool replace` eingeben zu müssen.

  Die Eigenschaft `autoreplace` legt die Charakteristika des automatischen Erkennens ausgewechselter Geräte fest. Wenn diese Eigenschaft auf `off` gesetzt ist, muss das Auswechseln von Speichergeräten vom Administrator mithilfe des Befehls `zpool replace` initiiert werden. Wenn diese Eigenschaft auf `on` gesetzt ist, wird jedes neue Speichergerät, das an derselben Speicherstelle erkannt wird, an der zuvor ein andres Speichergerät im Pool vorhanden war, automatisch formatiert und in den Pool eingebunden. Das Standardverhalten ist `off`
- Wenn ein Speichergerät oder Hot-Spare bei laufendem Betrieb physisch entfernt wurde, steht jetzt der Speicherpoolstatus `REMOVED` zur Verfügung. Falls verfügbar, wird ein Hot-Spare für das entfernte Speichergerät in den Pool eingebunden.

- Wenn ein Speichergerät entfernt und danach wieder eingesetzt wird, wird es in Betrieb genommen. Wenn für das entfernte Speichergerät ein Hot-Spare eingebunden wurde, wird dieses nach Abschluss der Inbetriebnahme wieder entfernt.
- Das automatische Erkennen entfernter und hinzugefügter Speichergeräte ist hardwareabhängig und wird nicht von allen Plattformen unterstützt. So werden USB-Speichergeräte beispielsweise beim Einfügen automatisch konfiguriert. SATA-Laufwerke müssen unter Umständen jedoch mit dem Befehl `cfgadm -c configure` konfiguriert werden.
- Hot-Spares werden regelmäßig überprüft, um sicherzustellen, dass sie in Betrieb und verfügbar sind.

Weitere Informationen finden Sie in der Manpage zpool(1M).

Rekursives Umbenennen von ZFS-Snapshots (zfs rename -r)-

Solaris 10 10/08: Der Befehl `zfs rename -r` ermöglicht es, alle untergeordneten ZFS-Snapshots rekursiv umzubenennen. Beispiel:

Zunächst wird ein Snapshot einiger ZFS-Dateisysteme erstellt.

```
# zfs snapshot -r users/home@today
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
users                       216K  16.5G    20K  /users
users/home                   76K  16.5G    22K  /users/home
users/home@today              0      -     22K  -
users/home/markm             18K  16.5G    18K  /users/home/markm
users/home/markm@today        0      -     18K  -
users/home/marks             18K  16.5G    18K  /users/home/marks
users/home/marks@today        0      -     18K  -
users/home/neil              18K  16.5G    18K  /users/home/neil
users/home/neil@today         0      -     18K  -
```

Am folgenden Tag werden die Snapshots dann umbenannt.

```
# zfs rename -r users/home@today @yesterday
# zfs list
NAME                            USED  AVAIL  REFER  MOUNTPOINT
users                           216K  16.5G    20K  /users
users/home                       76K  16.5G    22K  /users/home
users/home@yesterday              0      -     22K  -
users/home/markm                 18K  16.5G    18K  /users/home/markm
users/home/markm@yesterday        0      -     18K  -
users/home/marks                 18K  16.5G    18K  /users/home/marks
users/home/marks@yesterday        0      -     18K  -
users/home/neil                  18K  16.5G    18K  /users/home/neil
users/home/neil@yesterday         0      -     18K  -
```

Snapshots sind die einzigen Datasets, die rekursiv umbenannt werden können.

Weitere Informationen zu Snapshots finden Sie unter „Überblick über ZFS-Snapshots“ auf Seite 225 und in folgendem Blog-Eintrag, in dem die Erstellung rotierender Snapshots beschrieben ist:

http://blogs.sun.com/mmusante/entry/rolling_snapshots_made_easy

Für ZFS ist gzip-Komprimierung verfügbar

Solaris 10 10/08: In dieser Solaris-Version können ZFS-Dateisysteme zusätzlich zu lzjb auch mit gzip komprimiert werden. Sie können die Komprimierung mit gzip oder gzip-*N* angeben, wobei *N* den Wert 1 bis 9 haben kann. Beispiel:

```
# zfs create -o compression=gzip users/home/snapshots
# zfs get compression users/home/snapshots
NAME                  PROPERTY     VALUE            SOURCE
users/home/snapshots  compression  gzip             local
# zfs create -o compression=gzip-9 users/home/oldfiles
# zfs get compression users/home/oldfiles
NAME                  PROPERTY     VALUE           SOURCE
users/home/oldfiles   compression  gzip-9          local
```

Weitere Informationen zum Setzen von ZFS-Eigenschaften finden Sie unter „Setzen von ZFS-Eigenschaften“ auf Seite 206.

Speichern mehrerer Kopien von ZFS-Benutzerdaten

Solaris 10 10/08: Sofern möglich, werden Metadaten des ZFS-Dateisystems aus Gründen der Zuverlässigkeit automatisch auf mehreren Festplatten gespeichert. Dies wird als *ditto blocks* bezeichnet.

In dieser Version können Sie über den Befehl zfs set copies festlegen, dass mehrere Kopien der Benutzerdaten auch pro Dateisystem gespeichert werden. Beispiel:

```
# zfs set copies=2 users/home
# zfs get copies users/home
NAME        PROPERTY  VALUE       SOURCE
users/home  copies    2           local
```

Verfügbare Werte sind 1, 2 oder 3. Der Standardwert ist 1. Diese Kopien werden zusätzlich zu den von Redundanzfunktionen (Datenspiegelung bzw. RAID-Z) auf Pool-Ebene angelegten Sicherungskopien erstellt.

Die Speicherung mehrerer Kopien von ZFS-Benutzerdaten bringt die folgenden Vorteile mit sich:

- Verbesserte Datenaufbewahrung, da für alle ZFS-Konfigurationen die Wiederherstellung von nicht wiederherstellbaren Blocklesefehlern zugelassen wird, z. B. Datenträgerfehler (allgemein bekannt als *bit rot*)

- Schutz der Daten, auch wenn nur ein Laufwerk verfügbar ist
- Möglichkeit der Auswahl von Datenschutzrichtlinien auf Dateisystembasis jenseits der Grenzen des Speicher-Pools

---

Hinweis – Je nach Zuweisung der ditto blocks im Speicher-Pool ist es möglich, dass mehrere Kopien auf einer einzigen Festplatte abgelegt werden. Ein Ausfall einer ganzen Festplatte kann folglich bedeuten, dass sämtliche ditto blocks unverfügbar sind.

Die Verwendung von ditto blocks kann hilfreich sein, wenn Sie versehentlich einen nicht-redundanten Pool erstellen und Richtlinien für die Datenaufbewahrung festlegen müssen.

In folgendem Blog wird ausführlich beschrieben, wie sich das Speichern mehrerer Kopien in einem System mit Einzelplatten-Pool oder Mehrplatten-Pool auf den Datenschutz im Allgemeinen auswirken kann:

`http://blogs.sun.com/relling/entry/zfs_copies_and_data_protection`

Weitere Informationen zum Setzen von ZFS-Eigenschaften finden Sie unter „Setzen von ZFS-Eigenschaften“ auf Seite 206.

Verbesserte Ausgabe von `zpool status`

Solaris 10 8/07: Mit dem Befehl `zpool status -v` können Sie eine Liste der Dateien mit permanenten Fehlern ausgeben lassen. Bisher mussten die Dateinamen mit Hilfe des Befehls `find -inum` anhand der Liste der angezeigten Knoten ermittelt werden.

Weitere Informationen zum Anzeigen einer Liste von Dateien mit dauerhaften Fehlern finden Sie unter „Reparatur beschädigter Dateien bzw. Verzeichnisse“ auf Seite 323.

Verbesserungen für ZFS mit Solaris iSCSI

Solaris 10 8/07: In dieser Solaris-Version können Sie durch Setzen der Eigenschaft `shareiscsi` im ZFS-Volume ein ZFS-Volume als ein Solaris iSCSI-Zielgerät erstellen. Mithilfe dieses Verfahrens können Solaris iSCSI-Zielgeräte schnell eingerichtet werden. Beispiel:

```
# zfs create -V 2g tank/volumes/v2
# zfs set shareiscsi=on tank/volumes/v2
# iscsitadm list target
Target: tank/volumes/v2
    iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
    Connections: 0
```

Nach dem Erstellen des iSCSI-Zielgeräts können Sie den iSCSI-Initiator definieren. Informationen zur Einrichtung eines Solaris iSCSI-Initiators finden Sie in Kapitel 14, „Configuring Oracle Solaris iSCSI Targets and Initiators (Tasks)“ in *System Administration Guide: Devices and File Systems*.

Weitere Informationen zur Verwaltung eines ZFS-Volumes als iSCSI-Zielgerät entnehmen Sie bitte dem Abschnitt „Verwendung von ZFS-Volumes als Solaris-iSCSI-Zielgerät“ auf Seite 289.

ZFS-Befehlsprotokoll (`zpool history`)

Solaris 10 8/07: In dieser Solaris-Version protokolliert ZFS automatisch erfolgreich ausgeführte Aufrufe der Befehle `zfs` und `zpool`, die zur Änderung von Pool-Zustandsdaten dienen. Beispiel:

```
# zpool history
History for 'newpool':
2007-04-25.11:37:31 zpool create newpool mirror c0t8d0 c0t10d0
2007-04-25.11:37:46 zpool replace newpool c0t10d0 c0t9d0
2007-04-25.11:38:04 zpool attach newpool c0t9d0 c0t11d0
2007-04-25.11:38:09 zfs create newpool/user1
2007-04-25.11:38:15 zfs destroy newpool/user1

History for 'tank':
2007-04-25.11:46:28 zpool create tank mirror c1t0d0 c2t0d0 mirror c3t0d0 c4t0d0
```

Dank dieses Leistungsmerkmals können Sie oder Oracle-Supportmitarbeiter feststellen, welche ZFS-Befehle bei der Behebung eines Fehlers ausgeführt wurden.

Spezifische Speicher-Pools können mit dem Befehl `zpool history` identifiziert werden. Beispiel:

```
# zpool history newpool
History for 'newpool':
2007-04-25.11:37:31 zpool create newpool mirror c0t8d0 c0t10d0
2007-04-25.11:37:46 zpool replace newpool c0t10d0 c0t9d0
2007-04-25.11:38:04 zpool attach newpool c0t9d0 c0t11d0
2007-04-25.11:38:09 zfs create newpool/user1
2007-04-25.11:38:15 zfs destroy newpool/user1
```

In diesem Solaris-Release werden`user-ID`, *hostname* und *zone-name* über den Befehl *zpool history* nicht aufgezeichnet. Diese Informationen werden jedoch erst ab Solaris-Version 10 10/08 aufgezeichnet. Weitere Informationen finden Sie unter „Verbesserungen des ZFS-Befehlsprotokolls (`zpool history`)“ auf Seite 32.

Weitere Informationen zur Behebung von ZFS-Problemen siehe „Beheben von Problemen mit ZFS“ auf Seite 303.

Verbesserungen der ZFS-Eigenschaften

ZFS-Eigenschaft xattr

Solaris 10 8/07: Mit der Eigenschaft xattr können Sie erweiterte Attribute für ein bestimmtes ZFS-Dateisystem deaktivieren oder aktivieren. Der Standardwert ist on. Eine Beschreibung von ZFS-Eigenschaften finden Sie unter „ZFS-Eigenschaften“ auf Seite 189.

ZFS-Eigenschaft canmount

Solaris 10 8/07: Mit der neuen Eigenschaft canmount können Sie festlegen, ob ein Dataset mithilfe des Befehls zfs mount eingehängt werden kann. Weitere Informationen dazu finden Sie unter „Die Eigenschaft canmount“ auf Seite 201.

Benutzerdefinierte ZFS-Eigenschaften

Solaris 10 8/07: Zusätzlich zu den nativen Standardeigenschaften, die zum Exportieren interner Statistiken oder Steuern des ZFS-Dateisystemverhaltens dienen, stellt ZFS benutzerdefinierte Eigenschaften bereit. Benutzerdefinierte Eigenschaften wirken sich nicht auf das ZFS-Verhalten aus, können jedoch zum Versehen von Datasets mit Informationen, die für Ihre lokalen Gegebenheiten wichtig sind, verwendet werden.

Weitere Informationen finden Sie unter „Benutzerdefinierte ZFS-Eigenschaften“ auf Seite 202.

Setzen von Eigenschaften beim Erstellen von ZFS-Dateisystemen

Solaris 10 8/07: In dieser Solaris-Version können Sie Eigenschaften nicht nur nach, sondern auch bereits während der Erstellung von Dateisystemen festlegen.

Die folgenden Beispiele zeigen die entsprechende Syntax:

```
# zfs create tank/home
# zfs set mountpoint=/export/zfs tank/home
# zfs set sharenfs=on tank/home
# zfs set compression=on tank/home
```

```
# zfs create -o mountpoint=/export/zfs -o sharenfs=on -o compression=on tank/home
```

Anzeigen aller ZFS-Dateisysteminformationen

Solaris 10 8/07: In dieser Solaris-Version können Sie sich mithilfe verschiedener Formen des Befehls zfs get Informationen zu allen Datasets anzeigen lassen, wenn weder ein Dataset noch all angegeben wurde. Bisher war es nicht möglich, mit dem Befehl zfs get Informationen aller Datensätze anzuzeigen.

Beispiel:

```
# zfs get -s local all
tank/home               atime           off                     local
tank/home/bonwick       atime           off                     local
tank/home/marks         quota           50G                     local
```

Neue Option F für -zfs receive

Solaris 10 8/07: Sie können nun die neue Option -F für den Befehl zfs receive verwenden, um das Dateisystem auf den letzten Snapshot vor dem Empfang zurückzusetzen. Die Verwendung dieser Option kann erforderlich werden, wenn das Dateisystem nach einer Rücksetzung, aber vor Beginns des Empfangs geändert wurde.

Weitere Informationen finden Sie unter „Empfangen von ZFS-Snapshots" auf Seite 237.

Rekursive ZFS-Snapshots

Solaris 10 11/06: Wenn Sie mithilfe des Befehls zfs snapshot einen Dateisystem-Snapshot erstellen, können Sie die Option -r verwenden, um für alle untergeordneten Dateisysteme rekursiv Snapshots zu erstellen. Außerdem können Sie die Option - r verwenden, um alle untergeordneten Snapshots zu löschen, wenn ein Snapshot gelöscht wird.

Rekursive ZFS-Snapshots werden schnell in einem unteilbaren Vorgang erstellt. Snapshots werden entweder zusammen (d. h. alle auf einmal) oder gar nicht erstellt. Der Vorteil eines solchen Vorgangs besteht darin, dass die Snapshot-Daten stets zu einem einzigen konsistenten Zeitpunkt erstellt werden, selbst bei untergeordneten Dateisystemen.

Weitere Informationen finden Sie unter „Erstellen und Löschen von ZFS-Snapshots" auf Seite 226.

RAID-Z-Konfiguration mit doppelter Parität (raidz2)

Solaris 10 11/06: Redundante RAID-Z-Konfigurationen können jetzt einfache oder doppelte Parität besitzen. Das bedeutet, dass in einem System bis zu zwei Geräteausfälle ohne Datenverlust möglich sind. Eine RAID-Z-Konfiguration doppelter Parität kann mithilfe des Schlüsselworts raidz2 angegeben werden. Entsprechend können Sie für eine RAID-Z-Konfiguration mit einfacher Parität eines der Schlüsselwörter raidz oder raidz1 angeben.

Weitere Informationen finden Sie unter „Erstellen eines RAID-Z-Speicher-Pools" auf Seite 74 oder zpool(1M).

Hot-Spares für ZFS-Speicher-Pools

Solaris 10 11/06: Mithilfe der ZFS-Hot-Spare-Funktion können Sie Datenträger ermitteln, die zum Ersetzen eines ausgefallenen oder fehlerhaften Geräts in einem oder mehreren Speicher-Pools verwendet werden können. Das Vorsehen eines Datenträgers als *Hot-Spare*-Gerät bedeutet, dass bei Ausfall eines aktiven Datenträgers im Pool das Hot-Spare-Gerät diesen automatisch ersetzt. Alternativ dazu können Sie Datenträger in einem Speicher-Pool auch manuell durch ein Hot-Spare-Gerät ersetzen.

Weitere Informationen finden Sie unter „Zuweisen von Hot-Spares im Speicher-Pool“ auf Seite 98 und `zpool`(1M).

Ersetzen eines ZFS-Dateisystems durch einen ZFS-Klon (`zfs promote`)

Solaris 10 11/06: Der Befehl `zfs promote` ermöglicht es, ein vorhandenes ZFS-Dateisystem durch einen Klon desselben zu ersetzen. Diese Funktion ist hilfreich, wenn Sie an verschiedenen Versionen eines Dateisystems Tests ausführen wollen und danach eine alternative Version des Dateisystems zum aktiven Dateisystem machen möchten.

Weitere Informationen finden Sie in „Ersetzen eines ZFS-Dateisystems durch einen ZFS-Klon“ auf Seite 233 und `zfs`(1M).

Aktualisieren von ZFS-Speicher-Pools (`zpool upgrade`)

Solaris 10 6/06: Mit dem Befehl `zpool upgrade` können Sie eine Aktualisierung eines Speicher-Pools vornehmen, um die neuesten Funktionen zu nutzen. Außerdem wurde der Befehl `zpool status` so geändert, dass Sie jetzt darauf hingewiesen werden, wenn Pools mit älteren ZFS-Versionen laufen.

Weitere Informationen finden Sie unter „Aktualisieren von ZFS-Speicher-Pools“ auf Seite 122 und `zpool`(1M).

Wenn Sie die ZFS-Administrationskonsole auf einem System mit einem Pool aus früheren Solaris-Versionen nutzen möchten, müssen Sie die Pools vor Verwendung der ZFS-Administrationskonsole aktualisieren. Mit dem Befehl `zpool status` lässt sich feststellen, ob Pools aktualisiert werden müssen. Informationen zur ZFS-Administrationskonsole finden Sie unter „Webbasierte ZFS-Verwaltung“ auf Seite 45.

ZFS-Befehle ?backup“ und ?restore“ wurden umbenannt

Solaris 10 6/06: In dieser Solaris-Version heißen die Befehle `zfs backup` und `zfs restore` nun `zfs send` und `zfs receive`. Diese Namen geben die Funktion der Befehle genauer wieder. Diese Befehle dienen zum Senden und Empfangen von ZFS-Datenstromobjekten.

Weitere Informationen zu diesen Befehlen finden Sie unter „Senden und Empfangen von ZFS-Daten“ auf Seite 234.

Wiederherstellen gelöschter Speicher-Pools

Solaris 10 6/06: Diese Solaris-Version enthält den Befehl `zpool import -D`. Damit können Sie Pools wiederherstellen, die zuvor mit dem Befehl `zpool destroy` gelöscht wurden.

Weitere Informationen dazu finden Sie unter „Wiederherstellen gelöschter ZFS-Speicher-Pools“ auf Seite 120.

Integration von ZFS mit Fault Manager

Solaris 10 6/06: Diese Version enthält ein integriertes ZFS-Diagnoseprogramm, das Pool- und Datenträgerausfälle diagnostiziert und meldet. Darüber hinaus werden auch mit solchen Pool- bzw. Datenträgerausfällen im Zusammhang stehende Prüfsummen-, E/A-, Geräte- und Poolfehler gemeldet.

Das Diagnoseprogramm enthält keine Funktion zur Früherkennung von Prüfsummen- bzw. E/A-Fehlern und umfasst auch keine auf Fehleranalysen beruhenden proaktiven Operationen.

Bei einem ZFS-Ausfall wird eine Meldung wie die folgende angezeigt:

```
SUNW-MSG-ID: ZFS-8000-D3, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Wed Jun 30 14:53:39 MDT 2010
PLATFORM: SUNW,Sun-Fire-880, CSN: -, HOSTNAME: neo
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: 504a1188-b270-4ab0-af4e-8a77680576b8
DESC: A ZFS device failed.  Refer to http://sun.com/msg/ZFS-8000-D3 for more information.
AUTO-RESPONSE: No automated response will occur.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Run ’zpool status -x’ and replace the bad device.
```

Durch Überprüfen der empfohlenen Aktion, die nach den spezifischeren Anweisungen im Befehl `zpool status` angezeigt wird, können Sie die Fehlerursache schnell erkennen und beheben.

Ein Beispiel für die Wiederherstellung des Normalbetriebs nach einem aufgetretenen ZFS-Problem finden Sie unter „Abhilfe bei Nichtverfügbarkeit eines Geräts“ auf Seite 309.

Der Befehl `zpool clear`

Solaris 10 6/06: Diese Version enthält den Befehl `zpool clear`, mit dem Fehlerzähler für Geräte- bzw. Poolausfälle zurückgesetzt werden können. In früheren Versionen wurden Fehlerzähler bei der Wiederinbetriebnahme eines Datenträgers im Pool mithilfe des Befehls `zpool online` zurückgesetzt. Weitere Informationen finden Sie unter „Löschen von Gerätefehlern im Speicher-Pool“ auf Seite 95 und `zpool`(1M).

Kompaktes Format von NFSv4-Zugriffssteuerungslisten

Solaris 10 6/06: In dieser Version können Sie NFSv4-Zugriffssteuerungslisten in zwei Formaten setzen und anzeigen: ausführlich und kompakt. Mit dem Befehl `chmod` können Sie alle Zugriffssteuerungslistenformate setzen. Mit dem Befehl `ls - V` können Sie das kompakte Zugriffssteuerungslistenformat anzeigen. Mit dem Befehl `ls -v` können Sie das ausführliche Zugriffssteuerungslistenformat anzeigen.

Weitere Informationen finden Sie unter „Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im Kompaktformat“ auf Seite 266, `chmod`(1) und `ls`(1).

Dienstprogramm `fsstat` zum Überwachen von Dateisystemen

Solaris 10 6/06: Zum Melden von Vorgängen in Dateisystemen steht das neue Tool `fsstat` zur Verfügung. Aktivitäten können nach Einhängepunkt oder Dateisystemtypen protokolliert werden. Das folgende Beispiel zeigt eine allgemeine Aktivität eines ZFS-Dateisystems:

```
$ fsstat zfs
 new  name   name  attr  attr lookup rddir  read read  write write
 file remov  chng   get   set    ops   ops   ops bytes   ops bytes
7.82M 5.92M 2.76M 1.02G 3.32M  5.60G 87.0M  363M 1.86T 20.9M  251G zfs
```

Weitere Informationen finden Sie in der Manpage `fsstat`(1M).

Webbasierte ZFS-Verwaltung

Solaris 10 6/06: Mit einem webbasierten ZFS-Verwaltungstool, der ZFS-Administrationskonsole, können Sie folgende Verwaltungsaufgaben ausführen:

- Erstellen eines neuen Datenspeicher-Pools
- Erweitern der Kapazität eines vorhandenen Datenspeicher-Pools
- Verlagern (Exportieren) eines Datenspeicher-Pools auf ein anderes System

- Importieren eines zuvor exportierten Datenspeicher-Pools, um diesen auf einem anderen System verfügbar zu machen
- Anzeigen von Informationen zu Datenspeicher-Pools
- Erstellen von Dateisystemen
- Erstellen von Volumes
- Erstellen eines Snapshots eines Dateisystems oder Volumes
- Wiederherstellen eines früheren Snapshots eines Dateisystems

Sie können mithilfe eines sicheren Webbrowsers unter der folgenden URL auf die ZFS-Administrationskonsole zugreifen:

`https://`*system-name*`:6789/zfs`

Wenn Sie die entsprechende URL eingeben und die ZFS-Administrationskonsole nicht erreichen, kann es sein, dass deren Server nicht läuft. Geben Sie den folgenden Befehl ein, um den Server zu starten:

```
# /usr/sbin/smcwebserver start
```

Geben Sie den folgenden Befehl ein, wenn der Server beim Hochfahren des Systems automatisch gestartet werden soll:

```
# /usr/sbin/smcwebserver enable
```

Hinweis – Die Solaris Management Console (`smc`) kann nicht zur Verwaltung von ZFS-Speicher-Pools bzw. -Dateisystemen verwendet werden.

Was ist ZFS?

Das ZFS-Dateisystem ist ein revolutionäres neues Dateisystem, das die Verwaltung von Dateisystemen grundlegend ändert. Es besitzt Leistungsmerkmale und Vorteile, die in keinem anderen heutzutage verfügbaren Dateisystem zu finden sind. ZFS ist robust, skalierbar und einfach zu verwalten.

Speicher-Pools in ZFS

Die physische Datenspeicherung beruht bei ZFS auf dem Konzept der *Speicher-Pools*. Früher wurden Dateisysteme auf ein einziges physisches Datenspeichergerät aufsetzend konzipiert. Damit mehrere Datenspeichergeräte adressiert werden können und Datenredundanz möglich wird, wurde das Konzept eines so genannten *Volume Manager* entwickelt, um für die Darstellung eines einzelnen Datenspeichergeräts zu sorgen. Dadurch müssen Dateisysteme

nicht modifiziert werden, wenn mehrere Datenspeichergeräte genutzt werden sollen. Dieses Konzept brachte mehr Komplexität mit sich und behinderte die Weiterentwicklung, da das Dateisystem keine Kontrolle über die physische Speicherung von Daten auf den virtualisierten Volumes hatte.

In ZFS ist die Verwaltung einzelner Volumes nicht mehr erforderlich. Anstatt einer erzwungenen Erstellung virtueller Volume-Systeme fasst ZFS Datenspeichergeräte in so genannten Speicher-Pools zusammen. Ein solcher Speicher-Pool bestimmt die physischen Eigenschaften der Speicherung (Gerätestruktur, Datenredundanz usw.) und fungiert als flexibler Datenspeicher, aus dem Dateisysteme erstellt werden können. Dateisysteme sind nicht mehr auf bestimmte Speichergeräte beschränkt und können die Festplattenkapazität im Pool gemeinsam nutzen. Sie müssen die Kapazität eines Dateisystems nicht mehr vorher festlegen, da die Dateisysteme automatisch innerhalb der Festplattenkapazität erweitert werden, die im Speicher-Pool verfügbar ist. Wenn ein Pool um neuen Speicherplatz erweitert wird, können alle Dateisysteme dieses Pools diese neue Festplattenkapazität sofort nutzen, ohne dass dafür Konfigurationen geändert werden müssen. In mancherlei Hinsicht lässt sich die Funktionsweise eines Speicher-Pools mit der eines virtuellen Speichersystems vergleichen: Wenn ein DIMM-Speicherchip in ein System eingebaut wird, werden Sie vom Betriebssystem nicht gezwungen, Befehle auszuführen, um den neuen Speicher zu konfigurieren und einzelnen Prozessen zuzuweisen. Alle Prozesse des Systems verwenden automatisch diesen zusätzlichen Speicherplatz.

Transaktionale Semantik

ZFS ist ein transaktionales Dateisystem. Das bedeutet, dass der Dateisystemstatus auf dem Datenträger stets konsistent ist. Bei herkömmlichen Dateisystemen werden vorhandene Daten überschrieben. Dies kann dazu führen, dass sich ein Dateisystem in einem instabilen Zustand befindet, wenn beispielsweise zwischen dem Zeitpunkt der Zuweisung eines Datenblocks und dem der Verknüpfung mit einem Verzeichnis ein Stromausfall aufgetreten war. Früher wurde dieses Problem durch den Befehl `fsck` behoben. Dieser Befehl diente dem Zweck, den Zustand des Dateisystems zu überprüfen und zu versuchen, Inkonsistenzen zu beheben. Das Problem von inkonsistenten Dateisystemen verursachte Systemadministratoren viele Unannehmlichkeiten, und der Befehl `fsck` bot nicht die Garantie, alle Probleme zu beheben. Zuletzt wurden Dateisysteme entwickelt, die auf dem Konzept des so genannten *Journaling* beruhen. *Beim Journaling werden Aktionen in einem eigenen ?Journal" festgehalten, das im Falle von Systemabstürzen sicher eingespielt werden kann.* Diese Methode verursacht jedoch unnötigen Datenverarbeitungsaufwand, da Daten zweimal geschrieben werden müssen. Außerdem entstehen oft weitere Probleme, wenn beispielsweise das Journal nicht fehlerfrei gelesen werden konnte.

Transaktionale Dateisysteme verwalten Daten mithilfe der so genannten *Copy on Write-Semantik*. Bei diesem Verfahren werden niemals Daten überschrieben, und jegliche Folge von Vorgängen wird entweder vollständig ausgeführt oder ganz ignoriert. Das bedeutet, dass ein Dateisystem bei diesem Verfahren durch Stromausfälle oder Systemabstürze

grundsätzlich nicht beschädigt werden kann. Obwohl ganz zuletzt gespeicherte Daten unter Umständen verloren gehen können, bleibt das Dateisystem selbst stets konsistent. Darüber hinaus werden (mithilfe des `O_DSYNC`-Flags geschriebene) synchrone Daten stets vor der Rückkehr geschrieben, sodass sie niemals verloren gehen.

Prüfsummen und Daten mit Selbstheilungsfunktion

In ZFS werden alle Daten und Metadaten durch einen vom Benutzer auswählbaren Prüfsummenalgorithmus verifiziert. Herkömmliche Dateisysteme führten die Prüfsummenverifizierung datenblockweise aus, was auf die Volume-Verwaltungsschicht und den Aufbau dieser Dateisysteme zurückzuführen war. Aufgrund des herkömmlichen Aufbaus dieser Dateisysteme kann es vorkommen, dass bestimmte Fehler wie z. B. das Speichern eines Datenblocks an einem falschen Speicherort dazu führen kann, dass trotz fehlerhafter Daten keine Prüfsummenfehler vorhanden sind. ZFS-Prüfsummen werden so gespeichert, dass solche Fehler erkannt und der ordnungsgemäße Zustand des Dateisystems wiederhergestellt werden kann. Alle Prüfsummenverifizierungen und Datenwiederherstellungen finden auf Dateisystemebene statt und sind so für Anwendungsprogramme sichtbar.

Darüber hinaus bietet ZFS Selbstheilungsfunktionen für Daten. ZFS unterstützt Speicher-Pools mit unterschiedlichen Stufen der Datenredundanz. Bei Erkennung beschädigter Datenblöcke ruft ZFS die unbeschädigten Daten von einer redundanten Kopie ab und repariert die beschädigten Daten, indem es diese durch korrekte Daten ersetzt.

Konkurrenzlose Skalierbarkeit

Ein Schlüsselelement des ZFS-Dateisystems ist Skalierbarkeit. Das Dateisystem beruht auf der 128-Bit-Architektur, was die Speicherung von 256 Billiarden Zettabyte Daten ermöglicht. Alle Metadata werden dynamisch zugewiesen. Damit entfällt die Notwendigkeit, Inodes vorher zuweisen bzw. die Skalierbarkeit bei der ersten Erstellung eines Dateisystems anderweitig einschränken zu müssen. Alle Algorithmen wurden im Hinblick auf eine bestmögliche Skalierbarkeit entwickelt. Verzeichnisse können bis zu 2^{48} (256 Billionen) Einträge enthalten, und für die Anzahl der Dateisysteme bzw. die Anzahl der in einem Dateisystem enthaltenen Dateien bestehen keine Beschränkungen.

ZFS-Snapshots

Ein *Snapshot* ist eine schreibgeschützte Kopie eines Dateisystems bzw. Volumes. Snapshots können schnell und einfach erstellt werden. Anfänglich belegen Snapshots keine zusätzliche Festplattenkapazität im Pool.

Wenn Daten innerhalb des aktiven Datasets geändert werden, belegt der Snapshot Festplattenkapazität, da die alten Daten weiterhin referenziert werden. So verhindern Snapshots, dass der von den Daten belegte Speicherplatz für den Pool freigegeben wird.

Vereinfachte Administration

Eine der wichtigsten Eigenschaften von ZFS ist das erheblich vereinfachte Administrationsmodell. Durch hierarchische Dateisystemstrukturen, Eigenschaftsvererbung sowie automatische Verwaltung von Einhängepunkten und NFS-Netzwerksemantik können ZFS-Dateisysteme auf einfache Weise erstellt und verwaltet werden, ohne dass dafür mehrere Befehle ausgeführt oder Konfigurationsdateien bearbeitet werden müssen. Sie können auf einfache Weise Kontingente bzw. Reservierungen vornehmen, Datenkomprimierung aktivieren/deaktivieren oder Einhängepunkte für mehrere Dateisysteme mit einem einzigen Befehl verwalten. Sie können Datenspeichergeräte überprüfen oder reparieren, ohne Kenntnisse über einen separaten Volume-Manager-Befehlssatz zu besitzen. Sie können Datenströme von Dateisystem-Snapshots senden und empfangen.

ZFS verwaltet Dateisysteme über eine Hierarchie, die eine vereinfachte Verwaltung von Eigenschaften wie z. B. Kontingenten, Reservierungen, Komprimierung und Einhängepunkten ermöglicht In diesem Modell werden Dateisysteme zur zentralen Verwaltungsstelle. Dateisysteme sind sehr kostengünstig (ähnlich wie die Erstellung eines neuen Verzeichnisses). Deswegen sollten Sie für jeden Benutzer, jedes Projekt, jeden Arbeitsbereich usw. ein neues Dateisystem erstellen. Dieses Konzept ermöglicht Ihnen, Verwaltungspunkte genau zu definieren.

In ZFS verwendete Begriffe

In diesem Abschnitt werden die im vorliegenden Dokument verwendeten Begriffe erläutert:

Alternative Boot-Umgebung
: Eine Boot-Umgebung, die mit dem Befehl `lucreate` erstellt und mit dem Befehl `luupgrade` aktualisiert werden kann. Bei dieser Boot-Umgebung handelt es sich aber weder um die aktive noch um die primäre Boot-Umgebung. Die alternative Boot-Umgebung kann mithilfe des Befehls `luactivate` zur primären Boot-Umgebung gemacht werden.

checksum
: Mithilfe eines 256-Bit-Hash-Algorithmus verschlüsselte Daten eines Dateisystemblocks. Prüfsummenalgorithmen reichen vom einfachen und schnellen Fletcher4-Algorithmus (die Standardeinstellung) bis hin zu Hash-Algorithmen mit starker Verschlüsselung wie z. B. SHA256.

Klon
: Ein Dateisystem, dessen anfänglicher Inhalt identisch mit dem Inhalt eines Snapshots ist.

: Informationen zu Klonen finden Sie in „Überblick über ZFS-Klone“ auf Seite 232.

Dataset
: Ein allgemeiner Name für die folgenden ZFS-Komponenten: Klone, Dateisysteme, Snapshots und Volumes.

Jedes Dataset wird im ZFS-Namensraum durch einen eindeutigen Namen identifiziert. Datasets werden mithilfe des folgenden Formats benannt:

Pool/*Pfad*[@*Snapshot*]

pool	Der Name des Speicher-Pools, der das Dataset enthält
Pfad	Ein durch Schrägstriche begrenzter Pfadname für die Dataset-Komponente
Snapshot	Eine optionale Komponente, die den Snapshot eines Datasets identifiziert

Weitere Informationen zu Datasets finden Sie in Kapitel 6, „Verwalten von Oracle Solaris ZFS-Dateisystemen“.

Dateisystem

Ein ZFS-Dataset der Art `filesystem`, der im Standardnamensraum des Systems eingehängt ist und sich wie jedes andere Dateisystem verhält.

Weitere Informationen zu Dateisystemen finden Sie in Kapitel 6, „Verwalten von Oracle Solaris ZFS-Dateisystemen“.

Mirror

Eine Konfiguration mit virtuellen Geräten, in der identische Kopien von Daten auf zwei oder mehr Festplatten gespeichert werden. Wenn eine Festplatte im Datenspiegelungssystem ausfällt, stellt eine andere Festplatte dieses Systems die gleichen Daten bereit.

Pool

Eine logische Gruppe von Geräten, die das Layout und die physischen Merkmale des verfügbaren Speichers beschreibt. Datensätzen wird Festplattenkapazität aus einem Pool zugewiesen.

Weitere Informationen zu Speicher-Pools finden Sie in Kapitel 4, „Verwalten von Oracle Solaris ZFS-Speicher-Pools“.

Primäre Boot-Umgebung

Eine Boot-Umgebung, aus der mit dem Befehl `lucreate` die alternative Boot-Umgebung erstellt wird. Standardmäßig ist die primäre Boot-Umgebung die aktuelle Boot-Umgebung. Dieses Standardverhalten kann mit `lucreate` und der Option `-s` außer Kraft gesetzt werden.

RAID-Z
: Ein virtuelles Gerät, das Daten und Paritäten auf mehreren Festplatten speichert. Weitere Informationen zu RAID-Z finden Sie unter „Speicher-Pools mit RAID-Z-Konfiguration“ auf Seite 69.

Resilvering
: Den Vorgang des Kopierens von Daten von einem Datenspeichergerät auf ein anderes Gerät nennt man *Resilvering*. Wenn beispielsweise ein Spiegelungsgerät ersetzt oder offline geschaltet wird, werden die Daten eines auf dem aktuellen Stand befindlichen Spiegelungsgeräts auf das neu wiederhergestellte Gerät kopiert. Dieser Vorgang heißt in herkömmlichen Datenträgermanagement-Produkten *Neusynchronisierung der Datenspiegelung*.

  Weitere Informationen zum ZFS-Resilvering finden Sie in „Anzeigen des Resilvering-Status“ auf Seite 320.

snapshot
: Eine schreibgeschützte Kopie eines Dateisystems oder Volumes zu einem bestimmten Zeitpunkt.

  Weitere Informationen zu Snapshots finden Sie in „Überblick über ZFS-Snapshots“ auf Seite 225.

Virtuelles Gerät
: Ein logisches Gerät in einem Pool. Dies kann ein physisches Datenspeichergerät, eine Datei oder ein Geräteverbund sein.

  Weitere Informationen zu virtuellen Geräten finden Sie unter „Anzeigen von Informationen zu virtuellen Geräten in Storage-Pools“ auf Seite 77.

volume
: Ein Dataset, das eine Blockeinheit darstellt. Beispielsweise lässt sich ein ZFS-Volume als Swap-Gerät erstellen.

  Weitere Informationen zu ZFS-Volumes finden Sie unter „ZFS-Volumes“ auf Seite 287.

Konventionen für das Benennen von ZFS-Komponenten

Jede ZFS-Komponente wie beispielsweise ein Dataset oder ein Pool muss nach den folgenden Regeln benannt werden:

- Eine Komponente darf nur alphanumerische Zeichen sowie die folgenden vier Sonderzeichen enthalten:
  - Unterstrich (_)
  - Bindestrich (-)

- Doppelpunkt (:)
- Punkt (.)

- Pool-Namen müssen mit einem Buchstaben beginnen. Dabei gelten folgende Ausnahmen:
  - Die Zeichenfolge `c[0-9]` ist am Namensbeginn nicht erlaubt.
  - Der Name `log` ist reserviert.
  - Namen, die mit `mirror`, `raidz` , `raidz1`, `raidz2`, `raidz3` oder `spare` beginnen, sind nicht erlaubt, da diese Namen reserviert sind.
  - Pool-Namen dürfen kein Prozentzeichen (`%`) enthalten.
- Dataset-Namen müssen mit einem alphanumerischen Zeichen beginnen.
- Dataset-Namen dürfen kein Prozentzeichen (`%`) enthalten.

Außerdem sind leere Komponenten unzulässig.



KAPITEL 2

Erste Schritte mit Oracle Solaris ZFS

Dieses Kapitel enthält schrittweise Anleitungen zum Einrichten einer einfache Oracle Solaris ZFS-Konfiguration. Wenn Sie dieses Kapitel vollständig durchgearbeitet haben, sollte Ihnen das Funktionsprinzip von ZFS-Befehlen klar sein, und Sie sollten einfache Pools und Dateisysteme erstellen können. Dieses Kapitel ist nicht als allumfassende Informationsquelle gedacht und enthält Verweise auf die nächsten Kapitel, die ausführlichere Informationen enthalten.

Dieses Kapitel enthält die folgenden Abschnitte:

Hardware- und Softwarevoraussetzungen und -Empfehlungen für ZFS

Überprüfen Sie vor dem Arbeiten mit der ZFS-Software die folgenden Anforderungen an und Empfehlungen für Hardware und Software:

- Verwenden Sie ein SPARC- oder x86-System, auf dem Solaris 10 6/06 oder eine höhere Version installiert ist.
- Der für einen Speicher-Pool mindestens benötigte Speicherplatz beträgt 64 MB. Die Mindestkapazität für Festplatten beträgt 128 MB.
- Der für die Installation eines Solaris-Systems mindestens erforderliche Speicherplatz beträgt 768 MB. Zum Erreichen einer optimalen ZFS-Leistung werden jedoch mindestens 1 GB Speicher oder mehr empfohlen.
- Für die Erstellung eines Festplattensystems mit Datenspiegelung verwenden Sie mehrere Controller.

Erstellen eines einfachen ZFS-Dateisystems

Die ZFS-Administration wurde unter Berücksichtigung größtmöglicher Einfachheit entwickelt. Eines der konzeptionellen Ziele besteht in der Verringerung der Anzahl der Befehle, die zum Erstellen eines funktionierenden Dateisystems erforderlich sind. Beispielsweise wird beim Erstellen eines neuen Pools automatisch ein neues ZFS-Dateisystem erstellt und eingehängt.

Das folgende Beispiel zeigt, wie mit einem einzigen Befehl ein einfacher gespiegelter Speicher-Pool namens `tank` und ein ZFS-Dateisystem mit dem Namen `tank` erstellt werden können. Es wird angenommen, dass die Datenträger `/dev/dsk/c1t0d0` und `/dev/dsk/c2t0d0` vollständig verfügbar sind.

```
# zpool create tank mirror c1t0d0 c2t0d0
```

Weitere Informationen zu ZFS-Pool-Konfigurationen finden Sie unter „Replikationsfunktionen eines ZFS-Speicher-Pools" auf Seite 69.

Das neue ZFS-Dateisystem `tank` kann so viel Festplattenkapazität wie erforderlich belegen und wird automatisch unter `/tank` eingehängt.

```
# mkfile 100m /tank/foo
# df -h /tank
Filesystem             size   used  avail capacity  Mounted on
tank                   80G   100M    80G     1%    /tank
```

Innerhalb eines Pools werden Sie wahrscheinlich weitere Dateisysteme erstellen. Dateisysteme bieten Administrationsschnittstellen, mit deren Hilfe verschiedene Datengruppen innerhalb des gleichen Pools verwaltet werden können.

Das folgende Beispiel zeigt, wie ein Dateisystem namens `fs` im Speicher-Pool `tank` erstellt wird.

```
# zfs create tank/fs
```

Das neue ZFS-Dateisystem `tank/fs` kann so viel Festplattenkapazität wie erforderlich belegen und wird automatisch unter `/tank/fs` eingehängt.

```
# mkfile 100m /tank/fs/foo
# df -h /tank/fs
Filesystem             size   used  avail capacity  Mounted on
tank/fs                 80G   100M    80G     1%    /tank/fs
```

Sie werden wahrscheinlich eine Dateisystemhierarchie erstellen, die auf die Organisationsstruktur Ihrer Einrichtung abgestimmt ist. Weitere Informationen zum Erstellen einer Hierarchie von ZFS-Dateisystemen finden Sie unter „Erstellen einer ZFS-Dateisystemhierarchie" auf Seite 56.

Erstellen eines ZFS-Speicher-Pools

Das vorherige Beispiel verdeutlicht die Einfachheit von ZFS. Im Folgenden wird ein umfassenderes Beispiel vorgestellt, das einer praktischen Anwendung schon relativ nahe kommt. Zuerst müssen Sie Speicheranforderungen definieren und einen Speicher-Pool erstellen. Der Pool beschreibt die physischen Eigenschaften des Speicherbedarfs und muss vor dem Erstellen von Dateisystemen angelegt werden.

▼ So definieren Sie Speicheranforderungen für einen ZFS-Speicher-Pool

1 Legen Sie verfügbare Datenspeichergeräte für den Speicher-Pool fest.

Vor dem Erstellen eines Speicher-Pools müssen Sie festlegen, auf welchen Datenspeichergeräten Daten gespeichert werden sollen. Solche Datenspeichergeräte müssen eine Kapazität von mindestens 128 MB besitzen und dürfen nicht von anderen Bereichen des Betriebssystems verwendet werden. Die Datenspeichergeräte können einzelne Bereiche eines vorformatierten Datenträgers oder gesamte Datenträger sein, die ZFS als einzelnen größeren Bereich formatiert.

Im Speicherbeispiel unter „So erstellen Sie einen ZFS-Speicher-Pool" auf Seite 56 wird angenommen, dass die gesamten Datenträger `/dev/dsk/c1t0d0` und `/dev/dsk/c2t0d0` verfügbar sind.

Weitere Informationen zu Datenträgern und ihrer Verwendung und Bezeichnung finden Sie unter „Verwenden von Datenträgern in einem ZFS-Speicher-Pool" auf Seite 65.

2 Wählen Sie die Art der Datenreplikation.

ZFS unterstützt mehrere Datenreplikationsverfahren. Diese bestimmen, gegen welche Hardware-Ausfälle ein Pool gewappnet ist. ZFS unterstützt nicht redundante Konfigurationen (Stripes) sowie Datenspiegelung und RAID-Z (eine RAID-5-Variante).

Im Speicherbeispiel unter „So erstellen Sie einen ZFS-Speicher-Pool" auf Seite 56 wird eine einfache Datenspiegelung zweier verfügbarer Festplatten verwendet.

Weitere Informationen zur ZFS-Datenreplikation finden Sie unter „Replikationsfunktionen eines ZFS-Speicher-Pools" auf Seite 69.

▼ So erstellen Sie einen ZFS-Speicher-Pool

1. **Melden Sie sich als Root oder in einer gleichberechtigten Rolle (mithilfe des entsprechenden Zugriffsrechtsprofils von ZFS) an.**

   Weitere Informationen zu ZFS-Zugriffsrechtsprofilen finden Sie unter „ZFS-Zugriffsrechtsprofile" auf Seite 297.

2. **Legen Sie einen Namen für den Speicher-Pool fest.**

   Dieser Name dient zur Identifizierung des Speicher-Pools bei Verwendung der Befehle `zpool` und `zfs`. Da bei den meisten Systemen nur ein einziger Pool erstellt werden muss, können Sie einen beliebigen Namen auswählen. Beachten Sie dabei jedoch die unter „Konventionen für das Benennen von ZFS-Komponenten" auf Seite 51 aufgeführten Einschränkungen.

3. **Erstellen Sie den Pool.**

   Sie können beispielsweise mit dem folgenden Befehl einen Pool namens `tank` mit Datenspiegelung erstellen:

   ```
   # zpool create tank mirror c1t0d0 c2t0d0
   ```

   Wenn Datenspeichergeräte andere Dateisysteme enthalten oder anderweitig belegt sind, kann der Befehl den Pool nicht erstellen.

   Weitere Informationen zum Erstellen von Speicher-Pools finden Sie unter „Erstellen eines ZFS-Speicher-Pools" auf Seite 72. Weitere Informationen darüber, wie Sie herausfinden können, ob Datenspeichergeräte belegt sind oder nicht, finden Sie unter „Erkennen belegter Geräte" auf Seite 78.

4. **Überprüfen Sie die Ergebnisse.**

   Mit dem Befehl `zpool list` können Sie überprüfen, ob der Pool erfolgreich erstellt wurde.

   ```
   # zpool list
   NAME                    SIZE    ALLOC   FREE    CAP  HEALTH     ALTROOT
   tank                     80G    137K     80G     0%  ONLINE     -
   ```

   Weitere Informationen zum Anzeigen des Pool-Status finden Sie unter „Abfragen des Status von ZFS-Speicher-Pools" auf Seite 106.

Erstellen einer ZFS-Dateisystemhierarchie

Nach dem Erstellen eines Speicher-Pools zum Speichern von Daten können Sie mit dem Anlegen der Dateisystemhierarchie beginnen. Mit Hierarchien können Informationen auf einfache und gleichzeitig leistungsfähige Weise organisiert werden. Jedem, der bereits mit einem Dateisystem gearbeitet hat, sind diese vertraut.

In ZFS können Dateisysteme in Hierarchien organisiert werden, wobei jedes Dateisystem nur ein übergeordnetes System besitzt. Der Pool-Name ist stets der Name der obersten

Hierarchieebene. ZFS erweitert diese Hierarchie durch Eigenschaftsvererbung, sodass gemeinsame Eigenschaften schnell und einfach an gesamten Dateisystemhierarchien gesetzt werden können.

▼ So legen Sie eine ZFS-Dateisystemhierarchie fest

1 **Wählen Sie die Dateisystemgranularität aus.**

ZFS-Dateisysteme spielen für die Administration eine zentrale Rolle. Sie sind kompakt und können schnell erstellt werden. Eine gute Faustregel ist die Erstellung eines Dateisystems pro Benutzer bzw. Projekt, da ein solches Modell die benutzer- bzw. projektweise Kontrolle von Eigenschaften, Snapshots und Sicherungskopien ermöglicht.

Im Beispiel unter „So erstellen Sie ZFS-Dateisysteme" auf Seite 58 werden zwei ZFS-Dateisysteme (`bonwick` und `billm`) erstellt.

Weitere Informationen zur Verwaltung von Dateisystemen finden Sie in Kapitel 6, „Verwalten von Oracle Solaris ZFS-Dateisystemen".

2 **Fassen Sie ähnliche Dateisysteme zu Gruppen zusammen.**

In ZFS können Dateisysteme in Hierarchien organisiert werden, was die Gruppierung ähnlicher Dateisysteme ermöglicht. Dieses Modell bietet eine zentrale Administrationsschnittstelle zur Kontrolle von Eigenschaften und Verwaltung von Dateisystemen. Ähnliche Dateisysteme sollten unter einem gemeinsamen Namen erstellt werden.

Im Beispiel unter „So erstellen Sie ZFS-Dateisysteme" auf Seite 58 befinden sich die beiden Dateisysteme in einem Dateisystem namens `home`.

3 **Wählen Sie die Dateisystemeigenschaften.**

Die meisten Charakteristika von Dateisystemen werden mit Eigenschaften festgelegt. Diese Eigenschaften steuern verschiedene Verhaltensaspekte, so z. B. wo Dateisysteme eingehängt werden, ob und wie sie über das Netzwerk zugänglich sind, ob sie Datenkomprimierung verwenden und ob Kontingente gelten.

Im Beispiel unter „So erstellen Sie ZFS-Dateisysteme" auf Seite 58 sind alle home-Verzeichnisse unter `/export/zfs/` *user* eingehängt, mithilfe von NFS über das Netzwerk zugänglich und nutzen Datenkomprimierung. Außerdem wird für den Benutzer `bonwick` ein Kontingent von 10 GB erzwungen.

Weitere Informationen zu Eigenschaften finden Sie unter „ZFS-Eigenschaften" auf Seite 189.

▼ So erstellen Sie ZFS-Dateisysteme

1 Melden Sie sich als Root oder in einer gleichberechtigten Rolle (mithilfe des entsprechenden Zugriffsrechtsprofils von ZFS) an.

Weitere Informationen zu ZFS-Zugriffsrechtsprofilen finden Sie unter „ZFS-Zugriffsrechtsprofile" auf Seite 297.

2 Erstellen Sie die gewünschte Hierarchie.

In diesem Beispiel wird ein Dateisystem erstellt, das als Container für untergeordnete Dateisysteme dienen soll.

```
# zfs create tank/home
```

3 Legen Sie die vererbten Eigenschaften fest.

Nach dem Erstellen der Dateisystemhierarchie sollten Eigenschaften festgelegt werden, die für alle Benutzer gleich sind:

```
# zfs set mountpoint=/export/zfs tank/home
# zfs set sharenfs=on tank/home
# zfs set compression=on tank/home
# zfs get compression tank/home
NAME             PROPERTY       VALUE                      SOURCE
tank/home        compression    on                         local
```

Sie können Dateisystemeigenschaften beim Erstellen des Dateisystems festlegen. Beispiel:

```
# zfs create -o mountpoint=/export/zfs -o sharenfs=on -o compression=on tank/home
```

Weitere Informationen zu Eigenschaften und deren Vererbung finden Sie unter „ZFS-Eigenschaften" auf Seite 189.

Als Nächstes werden einzelne Dateisysteme im Dateisystem `home` des Pools `tank` gruppiert.

4 Erstellen Sie die einzelnen untergeordneten Dateisysteme.

Wären die Dateisysteme bereits erstellt, könnten die Eigenschaften anschließend auf der `home`-Ebene geändert werden. Alle Eigenschaften können dynamisch geändert werden, wenn Dateisysteme in Betrieb sind.

```
# zfs create tank/home/bonwick
# zfs create tank/home/billm
```

Diese Dateisysteme erben ihre Eigenschaftswerte von ihrem übergeordneten Dateisystem. Deswegen werden sie automatisch unter `/export/zfs/`*user* eingehängt und sind mit NFS über das Netzwerk zugänglich. Sie brauchen die Datei `/etc/vfstab` bzw. `/etc/dfs/dfstab` nicht zu bearbeiten.

Weitere Informationen zum Erstellen von Dateisystemen finden Sie unter „Erstellen eines ZFS-Dateisystems" auf Seite 186.

Weitere Informationen zum Einhängen von Dateisystemen und Freigeben von Dateisystemen für den Netzwerkzugang finden Sie in „Einhängen und Freigeben von ZFS-Dateisystemen“ auf Seite 211.

5 Legen Sie die dateisystemspezifischen Eigenschaften fest.

In diesem Beispiel ist dem Benutzer bonwick ein Kontingent von 10 GB zugewiesen. Diese Eigenschaft beschränkt unabhängig davon, wieviel Speicherplatz im Pool verfügbar ist, den durch diesen Benutzer zu belegenden Speicherplatz.

```
# zfs set quota=10G tank/home/bonwick
```

6 Überprüfen Sie die Ergebnisse.

Mit dem Befehl zfs list können verfügbare Dateisysteminformationen angezeigt werden:

```
# zfs list
NAME                    USED  AVAIL  REFER  MOUNTPOINT
tank                   92.0K  67.0G   9.5K  /tank
tank/home              24.0K  67.0G     8K  /export/zfs
tank/home/billm           8K  67.0G     8K  /export/zfs/billm
tank/home/bonwick         8K  10.0G     8K  /export/zfs/bonwick
```

Beachten Sie, dass der Benutzer bonwick maximal 10 GB Speicherplatz belegen darf, der Benutzer billm dagegen die gesamte Pool-Kapazität (67 GB) nutzen kann.

Weitere Informationen zum Anzeigen des Dateisystemstatus finden Sie unter „Abfragen von ZFS-Dateisysteminformationen“ auf Seite 204.

Weitere Informationen zur Belegung und Berechnung von Festplattenkapazität finden Sie unter „Berechnung von ZFS-Festplattenkapazität“ auf Seite 62.



Unterschiede zwischen Oracle Solaris ZFS und herkömmlichen Dateisystemen

In diesem Kapitel werden einige wichtige Unterschiede zwischen Oracle Solaris ZFS und herkömmlichen Dateisystemen erläutert. Durch das Verständnis dieser wichtigen Unterschiede werden Unklarheiten des Zusammenwirkens von herkömmlichen Tools mit ZFS ausgeräumt.

Dieses Kapitel enthält die folgenden Abschnitte:

Granularität von ZFS-Dateisystemen

Früher waren Dateisysteme auf ein Datenspeichergerät und damit auf die Kapazität dieses Geräts beschränkt. Das Erstellen und Neuerstellen herkömmlicher Dateisysteme aufgrund von Kapazitätsbeschränkungen ist zeitaufwändig und in einigen Fällen kompliziert. Herkömmliche Verwaltungslösungen unterstützen diesen Prozess.

Da ZFS-Dateisysteme nicht auf spezielle Geräte beschränkt sind, können sie ähnlich wie Verzeichnisse schnell und einfach erstellt werden. ZFS-Dateisysteme werden innerhalb der verfügbaren Festplattenkapazität des Speicher-Pools, in dem sie enthalten sind, automatisch vergrößert.

Anstatt ein Dateisystem (z. B. `/export/home`) zum Verwalten von Benutzerverzeichnissen zu erstellen, können Sie pro Benutzer ein Dateisystem anlegen. Sie können Dateisysteme auf einfache Weise einrichten und verwalten, indem Sie Eigenschaften festlegen, die von untergeordneten Dateisystemen innerhalb einer Hierarchie geerbt werden.

Ein Beispiel, das zeigt, wie eine Dateisystemhierarchie erstellt wird, finden Sie unter „Erstellen einer ZFS-Dateisystemhierarchie“ auf Seite 56.

Berechnung von ZFS-Festplattenkapazität

ZFS beruht auf dem Konzept der Datenspeicherung in Pools. Im Gegensatz zu herkömmlichen, physischen Datenträgern direkt zugewiesenen Dateisystemen teilen sich ZFS-Dateisysteme in einem Pool den in diesem Pool verfügbaren Speicherplatz. Somit kann sich die verfügbare Festplattenkapazität, die von Dienstprogrammen wie z. B. `df` gemeldet wird, auch dann ändern, wenn das betreffende Dateisystem inaktiv ist, da andere Dateisysteme im Pool Festplattenkapazität belegen bzw. freigeben.

Bitte beachten Sie, dass die maximal mögliche Dateisystemkapazität durch die Zuteilung von Kontingenten beschränkt werden kann. Weitere Informationen zu Kontingenten finden Sie unter „Setzen von Kontingenten für ZFS-Dateisysteme“ auf Seite 219. Durch Reservierungen kann einem Dateisystem eine bestimmte Festplattenkapazität garantiert werden. Weitere Informationen zu Reservierungen finden Sie unter „Setzen von Reservierungen für ZFS-Dateisysteme“ auf Seite 222. Dieses Modell gleicht dem NFS-Modell sehr; dort werden mehrere Verzeichnisse vom gleichen Dateisystem (z. B. `/home`) eingehängt.

Alle Metadaten werden in ZFS dynamisch zugewiesen. In herkömmlichen Dateisystemen erfolgt die Zuweisung von Metadaten meist vorher. Deswegen muss für diese Metadaten schon beim Erstellen des Dateisystems zusätzlicher Speicherplatz reserviert werden. Dieses Verhalten bedeutet weiterhin, dass die mögliche Gesamtdateianzahl eines Dateisystems schon vorher festgelegt ist. Da ZFS Metadaten nach Bedarf zuweist, muss vorher kein zusätzlicher Speicherplatz reserviert werden, und die Anzahl von Dateien wird lediglich durch die verfügbare Festplattenkapazität begrenzt. Die Ausgabe des Befehls `df -g` ist bei ZFS anders als bei herkömmlichen Dateisystemen zu interpretieren. Die unter `total files` ausgegebene Dateianzahl ist lediglich ein Schätzwert, der auf dem Betrag des im Pool verfügbaren Speicherplatzes beruht.

ZFS ist ein transaktionales Dateisystem. Das bedeutet, dass die meisten Dateisystemmodifikationen in Transaktionsgruppen zusammengefasst und asynchron auf dem Datenträger ausgeführt werden. Diese Modifikationen gelten so lange als *anstehende Änderungen*, bis sie auf dem Datenträger abgeschlossen sind. Die von einer Datei oder einem Dateisystem belegte, verfügbare und referenzierte Festplattenkapazität berücksichtigt keine anstehenden Änderungen. Anstehende Änderungen werden im Allgemeinen innerhalb weniger Sekunden abgeschlossen. Auch nach Abschluss eines Schreibvorgangs auf der Festplatte durch `fsync(3c)` oder `O_SYNC` werden die Informationen zur belegten Festplattenkapazität nicht unbedingt sofort aktualisiert.

Weitere Informationen zum Verbrauch von ZFS-Festplattenkapazität, der über die Befehle `du` und `df` gemeldet wird, finden Sie unter:

`http://hub.opensolaris.org/bin/view/Community+Group+zfs/faq/#whydusize`

Verhalten bei ungenügendem Speicherplatz

Snapshots von Dateisystemen sind in ZFS äußerst einfach und ohne hohen Aufwand zu erstellen. Snapshots sind in den meisten ZFS-Umgebungen vorhanden. Weitere Informationen zu ZFS-Snapshots finden Sie in Kapitel 7, „Arbeiten mit Oracle Solaris ZFS-Snapshots und -Klonen“.

Das Vorhandensein von Snapshots kann beim Freigeben von Festplattenkapazität ein unvorgesehenes Verhalten hervorrufen. Normalerweise können Sie mit den entsprechenden Zugriffsrechten eine Datei aus einem vollständigen Dateisystem löschen, wodurch im Dateisystem mehr Festplattenkapazität verfügbar wird. Wenn die zu löschende Datei jedoch in einem Snapshot eines Dateisystems vorhanden ist, wird durch das Löschen dieser Datei keine Festplattenkapazität freigegeben. Die von dieser Datei belegten Datenblöcke werden weiterhin vom Snapshot referenziert.

Deswegen kann infolge des Löschens einer Datei mehr Festplattenkapazität belegt werden, da eine neue Verzeichnisversion erstellt werden muss, die den neuen Namensraumstatus widerspiegelt. Dadurch können beim Löschen von Dateien unvorhergesehene `ENOSPC`- oder `EDQUOT`-Fehler auftreten.

Einhängen von ZFS-Dateisystemen

ZFS soll die Komplexität verringern und die Administration vereinfachen. Bei herkömmlichen Dateisystemen müssen Sie beispielsweise bei jedem Hinzufügen eines neuen Dateisystems die Datei `/etc/vfstab` entsprechend ändern. In ZFS ist das nicht mehr erforderlich, da Dateisysteme je nach den Dataset-Eigenschaften automatisch ein- und ausgehängt werden. ZFS-Einträge müssen also nicht in der Datei `/etc/vfstab` verwaltet werden.

Weitere Informationen zum Einhängen von Dateisystemen und Freigeben von ZFS-Dateisystemen für den Netzwerkzugang finden Sie unter „Einhängen und Freigeben von ZFS-Dateisystemen“ auf Seite 211.

Herkömmliche Datenträgerverwaltung

Wie bereits unter „Speicher-Pools in ZFS“ auf Seite 46 erwähnt, ist für ZFS kein spezielles Dienstprogramm zur Datenträgerverwaltung erforderlich. ZFS arbeitet mit im raw-Modus betriebenen Geräten. Deswegen können Speicher-Pools erstellt werden, die aus (software- oder hardwarebasierten) Logical Volumes bestehen. Diese Konfiguration wird jedoch nicht empfohlen, da ZFS am besten mit im raw-Modus betriebenen Geräten arbeitet. Die Verwendung von Logical Volumes kann sich negativ auf die Leistung und die Zuverlässigkeit auswirken und sollte deswegen vermieden werden.

Neues Solaris-Modell für Zugriffssteuerungslisten

Frühere Versionen des Betriebssystems Solaris unterstützten eine auf der POSIX-Spezifikation beruhende Implementierung von Zugriffssteuerungslisten. POSIX-basierte Zugriffssteuerungslisten dienen zum Schutz von UFS-Dateien. Zum Schutz von ZFS-Dateien wird ein neues, auf der NFSv4-Spezifikation beruhendes Solaris-Modell für Zugriffssteuerungslisten verwendet.

Die Hauptunterschiede des neuen Solaris-Zugriffssteuerungslistenmodells bestehen in Folgendem:

- Das Modell basiert auf der NFSv4-Spezifikation und ist den NT-Zugriffssteuerungslistenmodellen ähnlich.
- Das Modell bietet feiner abgestimmte Zugriffsrechte.
- Zugriffssteuerungslisten werden mit den Befehlen `chmod` und `ls` anstatt `setfacl` und `getfacl` eingestellt und angezeigt.
- Das Modell bietet eine reichhaltigere Vererbungssemantik zum Festlegen der Weitergabe von Zugriffsrechten von über- an untergeordnete Verzeichnisse usw.

Weitere Informationen zu Zugriffssteuerungslisten mit ZFS-Dateien finden Sie in Kapitel 8, „Schützen von Oracle Solaris ZFS-Dateien mit Zugriffssteuerungslisten“.



Verwalten von Oracle Solaris ZFS-Speicher-Pools

In diesem Kapitel wird beschrieben, wie Speicher-Pools in Oracle Solaris ZFS erstellt und verwaltet werden können.

Dieses Kapitel enthält die folgenden Abschnitte:

Komponenten eines ZFS-Speicher-Pools

Die folgenden Abschnitte enthalten ausführliche Informationen zu den folgenden Komponenten eines Speicher-Pools:

Verwenden von Datenträgern in einem ZFS-Speicher-Pool

Das grundlegendste Element eines Speicher-Pools ist physischer Speicher. Bei physischem Speicher kann es sich um eine Blockeinheit mit mindestens 128 MB Speicherkapazität handeln. Normalerweise handelt es sich dabei um Festplatten, die vom System im Verzeichnis `/dev/dsk` erkannt werden.

Ein Datenspeichergerät kann eine gesamte Festplatte (z. B. `c1t0d0`) oder ein einzelner Plattenbereich (z. B. `c0t0d0s7`) sein. Es wird empfohlen, eine gesamte Festplatte zu verwenden, da die Festplatte in diesem Fall nicht formatiert werden muss. ZFS formatiert solche Datenträger mit einem EFI-Label als einzelnen, großen Bereich. In diesem Fall wird die vom Befehl `format` ausgegebene Partitionstabelle in etwa wie folgt angezeigt:

```
Current partition table (original):
Total disk sectors available: 286722878 + 16384 (reserved sectors)

Part      Tag    Flag     First Sector         Size         Last Sector
  0        usr    wm               34      136.72GB          286722911
  1 unassigned    wm                0             0              0
  2 unassigned    wm                0             0              0
  3 unassigned    wm                0             0              0
  4 unassigned    wm                0             0              0
  5 unassigned    wm                0             0              0
  6 unassigned    wm                0             0              0
  8   reserved    wm        286722912        8.00MB          286739295
```

Um eine gesamte Festplatte verwenden zu können, muss diese der Benennungskonvention `/dev/dsk/cXtXdX` entsprechen. Einige Treiber von Drittanbietern nutzen andere Benennungskonventionen oder hängen Datenträger nicht im Verzeichnis `/dev/dsk`, sondern anderswo ein. Damit solche Datenträger verwendet werden können, müssen Sie diese manuell benennen und für ZFS einen Bereich verfügbar machen.

ZFS verwendet EFI-Label, wenn Sie Speicher-Pools mit gesamten Festplatten erstellen. Weitere Informationen zu EFI-Labels finden Sie unter „EFI Disk Label" in *System Administration Guide: Devices and File Systems*.

Eine für einen ZFS-Root-Pool gedachte Festplatte muss mit einem SMI-Label, nicht mit einem EFI-Label erstellt werden. Mit dem Befehl `format - e` können Sie eine Festplatte im Nachhinein mit einem SMI-Label versehen.

Datenträger können mit dem vollständigen Pfad (z. B. `/dev/dsk/c1t0d0` oder in einer Kurzschreibweise, die aus dem Gerätenamen im Verzeichnis `/dev/dsk` besteht (z. B. `c1t0d0`), angegeben werden. Im Folgenden sind beispielsweise gültige Datenträgernamen aufgeführt:

- `c1t0d0`
- `/dev/dsk/c1t0d0`
- `/dev/foo/disk`

Die Verwendung gesamter physischer Festplatten ist die einfachste Methode zum Erstellen von ZFS-Speicher-Pools. Was die Verwaltung, Zuverlässigkeit und Leistung angeht, so werden ZFS-Konfigurationen weitaus komplexer, wenn Sie Pools aus Datenträgerbereichen, LU-Nummern in Hardware-RAID-Arrays oder Volumes, die von Datenträgerverwaltungssoftware bereitgestellt werden, aufbauen. Bei der Entscheidung, ob Sie ZFS mit anderen hardware- bzw. softwarebasierten Speicherlösungen konfigurieren, sollten Sie die folgenden Aspekte berücksichtigen:

- Wenn Sie eine ZFS-Konfiguration mit LU-Nummern aus Hardware-RAID-Arrays aufbauen, muss Ihnen die Beziehung zwischen den Redundanzfunktionen von ZFS und denen des Arrays klar sein. Bestimmte Konfigurationen bieten ausreichende Redundanz und Leistung, bei anderen braucht dies aber nicht der Fall zu sein.
- Sie können logische Geräte für ZFS mithilfe von Volumes aufbauen, die von Software zur Datenträgerverwaltung wie z. B. Solaris Volume Manager (SVM) oder Veritas Volume Manager (VxVM) bereitgestellt werden. Solche Konfigurationen werden jedoch nicht empfohlen. Obwohl ZFS mit solchen Konfigurationen ordnungsgemäß funktioniert, ist die Leistung in diesen Fällen oftmals nicht optimal.

Zusätzliche Informationen zu Empfehlungen für Speicher-Pool-Konfigurationen finden Sie auf der „Best Practices"-Website für ZFS unter:

http://www.solarisinternals.com/wiki/index.php/ZFS_Best_Practices_Guide

Datenträger werden nach Pfad und Geräte-ID (falls verfügbar) identifiziert. Bei Systemen, in denen Geräte-ID-Informationen zur Verfügung stehen, erlaubt diese Identifizierungsmethode, Geräte neu zu konfigurieren, ohne dass eine Aktualisierung von ZFS nötig ist. Die Geräte-ID-Generierung und -Verwaltung kann von System zu System unterschiedlich sein. Darum sollten Sie den Pool zunächst exportieren und erst dann Geräte verschieben (beispielsweise eine Festplatte von einem Controller zu einem anderen). Ein Systemereignis wie beispielsweise eine Firmware-Aktualisierung oder eine andere Hardware-Änderung kann die Geräte-IDs im ZFS-Speicher-Pool verändern. Dadurch kann bewirkt werden, dass die Geräte nicht mehr zur Verfügung stehen.

Verwenden von Bereichen in einem ZFS-Speicher-Pool

Datenträger können mit einem herkömmlichen Solaris VTOC (SMI)-Label benannt werden, wenn Sie Speicher-Pools mit Festplattenbereichen erstellen.

Damit ein ZFS-Root-Pool boot-fähig ist, müssen die im Pool befindlichen Festplatten Bereiche enthalten und ein SMI-Label haben. Die einfachste Konfiguration würde darin bestehen, die gesamte Festplattenkapazität in Bereich 0 (Slice 0) zu legen und diesen Bereich für den Root-Pool zu verwenden.

In einem SPARC-System verfügt eine 72-GB-Festplatte über 68 GB nutzbaren Speicher im Bereich 0, wie in der folgenden Ausgabe von `format` zu sehen ist:

```
# format
.
.
.
Specify disk (enter its number): 4
selecting c1t1d0
partition> p
Current partition table (original):
```

```
Total disk cylinders available: 14087 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders         Size            Blocks
  0       root    wm       0 - 14086       68.35GB    (14087/0/0) 143349312
  1 unassigned    wm       0                0         (0/0/0)             0
  2     backup    wm       0 - 14086       68.35GB    (14087/0/0) 143349312
  3 unassigned    wm       0                0         (0/0/0)             0
  4 unassigned    wm       0                0         (0/0/0)             0
  5 unassigned    wm       0                0         (0/0/0)             0
  6 unassigned    wm       0                0         (0/0/0)             0
  7 unassigned    wm       0                0         (0/0/0)             0
```

In einem x86-System verfügt eine 72-GB-Festplatte über 68 GB nutzbaren Speicher im Bereich 0, wie in der folgenden Ausgabe von `format` zu sehen ist. Einige Boot-Infomationen sind in Bereich 8 enthalten. Bereich 8 erfordert keine Verwaltung und kann nicht verändert werden.

```
# format
.
.
.
selecting c1t0d0
partition> p
Current partition table (original):
Total disk cylinders available: 49779 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders         Size            Blocks
  0       root    wm       1 - 49778       68.36GB    (49778/0/0) 143360640
  1 unassigned    wu       0                0         (0/0/0)             0
  2     backup    wm       0 - 49778       68.36GB    (49779/0/0) 143363520
  3 unassigned    wu       0                0         (0/0/0)             0
  4 unassigned    wu       0                0         (0/0/0)             0
  5 unassigned    wu       0                0         (0/0/0)             0
  6 unassigned    wu       0                0         (0/0/0)             0
  7 unassigned    wu       0                0         (0/0/0)             0
  8       boot    wu       0 -     0        1.41MB    (1/0/0)          2880
  9 unassigned    wu       0                0         (0/0/0)             0
```

Verwenden von Dateien in einem ZFS-Speicher-Pool

In ZFS können Sie UFS-Dateien als virtuelle Geräte im Speicher-Pool verwenden. Dieses Merkmal dient vorrangig zum Testen und Aktivieren einfacher experimenteller Konfigurationen und ist nicht für den Praxiseinsatz unter Produktionsbedingungen gedacht. Dies ist darin begründet, dass **jede Verwendung von Dateien auf der Konsistenz des zugrunde liegenden Dateisystems beruht**. Wenn Sie einen ZFS-Pool mit Dateien aus einem UFS-Dateisystem erstellen, verlassen sie sich darauf, dass UFS ordnungsgemäße und synchrone Semantik gewährleistet.

Dateien können jedoch nützlich sein, wenn Sie ZFS zum ersten Mal testen oder mit komplexeren Konfigurationen experimentieren und nicht genügend physische Datenspeichergeräte zur Verfügung stehen. Alle Dateien müssen mit dem vollständigen Pfad angegeben werden und mindestens 64 MB groß sein.

Replikationsfunktionen eines ZFS-Speicher-Pools

ZFS bietet in Konfigurationen mit Datenspiegelung und RAID-Z Datenredundanz und Selbstheilungsfunktionen.

- „Speicher-Pools mit Datenspiegelung" auf Seite 69
- „Speicher-Pools mit RAID-Z-Konfiguration" auf Seite 69
- „Selbstheilende Daten in einer redundanten Konfiguration" auf Seite 71
- „Dynamisches Striping in einem Speicher-Pool" auf Seite 71
- „ZFS-Hybrid-Speicher-Pool" auf Seite 71

Speicher-Pools mit Datenspiegelung

Für Speicher-Pools mit Datenspiegelung sind mindestens zwei Datenträger erforderlich, auf die optimalerweise von zwei verschiedenen Controllern zugegriffen werden sollte. Für eine Datenspiegelung können viele Datenträger verwendet werden. Darüber hinaus können Sie in einem Pool mehrere Datenspiegelungen erstellen. Eine einfache Datenspiegelungskonfiguration würde im Prinzip ungefähr wie folgt aussehen:

```
mirror c1t0d0 c2t0d0
```

Eine komplexere Datenspiegelungskonfiguration würde im Prinzip ungefähr wie folgt aussehen:

```
mirror c1t0d0 c2t0d0 c3t0d0 mirror c4t0d0 c5t0d0 c6t0d0
```

Informationen zum Erstellen von Speicher-Pools mit Datenspiegelung finden Sie unter „Erstellen eines Speicher-Pools mit Datenspiegelung" auf Seite 72.

Speicher-Pools mit RAID-Z-Konfiguration

Neben Speicher-Pools mit Datenspiegelung bietet ZFS RAID-Z-Konfiguration, die Fehlertoleranzen mit ein-, zwei- oder dreifacher Parität aufweisen. RAID-Z mit einfacher Parität (`raidz` oder `raidz1`) ist mit RAID-5 vergleichbar. RAID-Z mit doppelter Parität (`raidz2`) ähnelt RAID-6.

Weitere Informationen über RAIDZ-3 (`raidz3`) finden Sie im folgenden Blog:

`http://blogs.sun.com/ahl/entry/triple_parity_raid_z`

Alle herkömmlichen Algorithmen, die RAID-5 ähnlich sind (z. B. RAID-4, RAID-5, RAID-6, RDP und EVEN-ODD), können von einem Problem betroffen sein, das als so genanntes ?RAID-5 Write Hole" bekannt ist. Wenn nur ein Teil eines RAID-5-Bereichs geschrieben wird und ein Stromausfall auftritt, bevor alle Datenblöcke auf den Datenträger geschrieben wurden,

kann die Parität der Daten nicht hergestellt werden, wodurch diese Daten nutzlos sind (es sei denn, sie werden von einem nachfolgenden vollständigen Bereich überschrieben). Bei RAID-Z verwendet ZFS RAID-Stripes veränderlicher Länge, sodass alle Schreibvorgänge vollständige Stripes speichern. Dieses Design ist nur möglich, weil ZFS die Verwaltung von Dateisystemen und Datenspeichergeräten so integriert, dass die Metadaten eines Dateisystems genügend Informationen zum zugrunde liegenden Datenredundanzmodell besitzen und sie somit RAID-Stripes veränderlicher Länge verarbeiten können. RAID-Z ist die weltweit erste nur auf Software basierte Lösung, die das RAID-5 Write Hole eliminiert.

Eine RAID-Z-Konfiguration mit N Datenträgern der Kapazität X mit P Paritätsdatenträgern kann ca. (N-P)*X Byte speichern und ist gegen einen Ausfall von P Geräten ohne Gefährdung der Datenintegrität geschützt. Für eine RAID-Z-Konfiguration mit einfacher Parität sind mindestens zwei, für eine RAID-Z-Konfiguration mit doppelter Parität mindestens drei Datenträger erforderlich. Wenn sich beispielsweise in einer RAID-Z-Konfiguration mit einfacher Parität drei Festplatten befinden, belegen Paritätsdaten eine Festplattenkapazität, die der Kapazität einer der drei Festplatten entspricht. Zum Erstellen einer RAID-Z-Konfiguration ist darüber hinaus keine spezielle Hardware erforderlich.

Eine RAID-Z-Konfiguration mit drei Datenträgern würde im Prinzip ungefähr wie folgt aussehen:

```
raidz c1t0d0 c2t0d0 c3t0d0
```

Eine komplexere RID-Z-Konfiguration würde im Prinzip ungefähr wie folgt aussehen:

```
raidz c1t0d0 c2t0d0 c3t0d0 c4t0d0 c5t0d0 c6t0d0 c7t0d0 raidz c8t0d0 c9t0d0 c10t0d0 c11t0d0
c12t0d0 c13t0d0 c14t0d0
```

Wenn Sie RAID-Z-Konfiguration mit mehreren Datenträgern erstellen, sollten die Datenträger in mehrere Gruppen aufgeteilt werden. Beispielsweise sollten Sie eine RAID-Z-Konfiguration mit 14 Datenträgern besser in zwei Gruppen mit je 7 Datenträgern aufteilen. RAID-Z-Konfigurationen mit weniger als zehn Datenträgern erreichen eine optimalere Leistung.

Informationen zum Erstellen eines RAID-Z-Speicher-Pools finden Sie unter „Erstellen eines RAID-Z-Speicher-Pools“ auf Seite 74.

Weitere Informationen zur Auswahl zwischen einer Datenspiegelungs- oder RAID-Z-Konfiguration, die auf Leistungs- und Festplattenkapazitätskriterien basiert, finden Sie im folgenden Blog:

`http://blogs.sun.com/roch/entry/when_to_and_not_to`

Zusätzliche Informationen zu Empfehlungen für RAID-Z-Speicher-Pools finden Sie auf der Best Practices-Website für ZFS unter:

`http://www.solarisinternals.com/wiki/index.php/ZFS_Best_Practices_Guide`

ZFS-Hybrid-Speicher-Pool

Der ZFS-Hybrid-Speicher-Pool, der zur Sun Storage 7000-Produktreihe von Oracle gehört, ist ein spezieller Speicher-Pool, der DRAM, SSDs und HDDs zur Erhöhung von Leistung und Kapazität bei gleichzeitiger Verringerung des Energieverbrauchs kombiniert. Sie können die ZFS-Redundanzkonfiguration des Speicher-Pools auswählen und einfach andere Konfigurationsoptionen mit dieser Verwaltungsschnittstelle des Produkts verwalten.

Weitere Informationen zu diesem Produkt finden Sie im *Sun Storage Unified Storage System Administration Guide* .

Selbstheilende Daten in einer redundanten Konfiguration

ZFS bietet in Konfigurationen mit Datenspiegelung und RAID-Z Selbstheilungsfunktionen.

Bei Erkennung beschädigter Datenblöcke ruft ZFS nicht nur die unbeschädigten Daten von einer redundanten Kopie ab, sondern repariert die beschädigten Daten auch, indem es diese mit der unbeschädigten Kopie ersetzt.

Dynamisches Striping in einem Speicher-Pool

ZFS verteilt Daten dynamisch mithilfe des Striping-Verfahrens über alle in der obersten Hierarchieebene befindlichen virtuellen Geräte. Die Entscheidung, wo Daten gespeichert werden, wird zur Zeit des Speichervorgangs getroffen.

Wenn neue virtuelle Geräte zu einem Pool hinzugefügt werden, weist ZFS dem neuen Gerät schrittweise Daten zu, wodurch Leistungsrichtlinien und Richtlinien für die Zuweisung von Festplattenkapazität eingehalten werden. Jedes virtuelle Gerät kann auch für die Datenspiegelung oder als RAID-Z-Gerät, das andere Datenträgergeräte bzw. Dateien enthält, eingesetzt werden. Mit dieser Konfiguration wird die Kontrolle der Fehlercharaketristika eines Pools flexibler. So können Sie beispielsweise aus vier Datenträgern die folgenden Konfigurationen bilden:

- vier Datenträger mit dynamischem Striping
- eine vierfache RAID-Z-Konfiguration
- zwei zweifache Geräte für die Datenspiegelung, die dynamisches Striping verwenden

Obwohl ZFS die Kombination verschiedener Arten virtueller Geräte im gleichen Pool unterstützt, wird dies nicht empfohlen. Sie können beispielsweise einen Pool mit einer zweifachen Datenspiegelungs- und einer dreifachen RAID-Z-Konfiguration erstellen. Die Fehlertoleranz des Gesamtsystems ist jedoch nur so gut wie die die Geräts mit der schlechtesten Fehlertoleranz, in diesem Fall RAID-Z. Eine bewährte Verfahrensweise ist, virtuelle Geräte der obersten Hierarchieebene zu verwenden, die das gleiche Redundanzniveau besitzen.

Erstellen und Entfernen von ZFS-Speicher-Pools

In den folgenden Abschnitten werden unterschiedliche Situationen zum Erstellen und Entfernen von ZFS-Speicher-Pools beschrieben:

- „Erstellen eines ZFS-Speicher-Pools" auf Seite 72
- „Anzeigen von Informationen zu virtuellen Geräten in Storage-Pools" auf Seite 77
- „Behandlung von Fehlern beim Erstellen von ZFS-Speicher-Pools" auf Seite 78
- „Löschen von ZFS-Speicher-Pools" auf Seite 81

Pools können schnell und einfach erstellt und entfernt werden. Allerdings sollten Sie solche Vorgänge äußerste Vorsicht durchführen. Obwohl Überprüfungen durchgeführt werden, die die Verwendung von Geräten verhindern, die bereits von einem anderen Pool benutzt werden, ist es ZFS nicht immer bekannt, ob ein Datenspeichergerät bereits belegt ist oder nicht. Das Entfernen eines Pools ist einfacher als die Erstellung eines Pools. Benutzen Sie den Befehl `zpool destroy` mit Vorsicht. Die Ausführung dieses einfachen Befehls kann bedeutende Konsequenzen nach sich ziehen.

Erstellen eines ZFS-Speicher-Pools

Speicher-Pools können mit dem Befehl `zpool create` erstellt werden. Dieser Befehl akzeptiert einen Pool-Namen und eine beliebige Anzahl virtueller Geräte als Argumente. Der Pool-Name muss den unter „Konventionen für das Benennen von ZFS-Komponenten" auf Seite 51 aufgeführten Namenskonventionen genügen.

Erstellen eines einfachen Speicher-Pools

Mit dem folgenden Befehl wird ein Pool namens `tank` erstellt, das aus den beiden Datenträgern `c1t0d0` und `c1t1d0` besteht:

```
# zpool create tank c1t0d0 c1t1d0
```

Gerätenamen, die ganze Datenträger repräsentieren, befinden sich im Verzeichnis `/dev/dsk` und werden von ZFS als einzelner, großer Bereich gekennzeichnet. Daten werden mithilfe von Striping dynamisch über beide Datenträger verteilt.

Erstellen eines Speicher-Pools mit Datenspiegelung

Verwenden Sie zum Erstellen eines Pools mit Datenspiegelung das Schlüsselwort `mirror`, gefolgt von einer bestimmten Anzahl von Datenspeichergeräten, aus denen das Datenspiegelungssystem bestehen soll Mehrere Datenspiegelungssysteme können durch Wiederholen des Schlüsselworts `mirror` in der Befehlszeile erstellt werden. Mit dem folgenden Befehl wird ein Pool mit zwei zweifachen Datenspiegelungen erstellt:

```
# zpool create tank mirror c1d0 c2d0 mirror c3d0 c4d0
```

Das zweite Schlüsselwort `mirror` spezifiziert ein neues virtuelles Gerät der obersten Hierarchieebene. Daten werden mithilfe von Striping dynamisch über beide Datenträger verteilt, wobei zwischen jedem Datenträger eine jeweilige Datenredundanz auftritt.

Weitere Informationen zu empfohlenen Konfigurationen mit Datenspiegelung finden Sie unter:

http://www.solarisinternals.com/wiki/index.php/ZFS_Best_Practices_Guide

Gegenwärtig werden für ZFS-Datenspiegelungskonfigurationen folgende Vorgänge unterstützt:

- Hinzufügen eines weiteren Festplattensatzes zu einer vorhandenen Datenspiegelungskonfiguration, um eine neues virtuelles Gerät der obersten Hierarchieebene bereitzustellen. Weitere Informationen dazu finden Sie unter „Hinzufügen von Datenspeichergeräten zu einem Speicher-Pool“ auf Seite 83.
- Hinzufügen weiterer Festplatten zu einer vorhandenen Datenspiegelungskonfiguration bzw. Hinzufügen weiterer Festplatten zu einer nicht replizierten Konfiguration, um eine Datenspiegelungskonfiguration zu erstellen. Weitere Informationen dazu finden Sie unter „Verbinden und Trennen von Geräten in einem Speicher-Pool“ auf Seite 87.
- Ersetzen einer oder mehrerer Festplatten in einer vorhandenen Datenspiegelungskonfiguration, solange die Kapazität der neuen Festplatten größer oder gleich der Kapazität der zu ersetzenden Festplatten ist. Weitere Informationen dazu finden Sie unter „Austauschen von Geräten in einem Speicher-Pool“ auf Seite 95.
- Abtrennen einer oder mehrerer Festplatten in einer Datenspiegelungskonfiguration, solange die verbleibenden Speichergeräte für die Konfiguration noch eine ausreichende Redundanz gewährleisten. Weitere Informationen dazu finden Sie unter „Verbinden und Trennen von Geräten in einem Speicher-Pool“ auf Seite 87.
- Teilen einer gespiegelten Konfiguration, indem eine der Festplatten entfernt wird, um einen neuen, identischen Pool zu erstellen. Weitere Informationen finden Sie unter „Erstellen eines neuen Pools durch Teilen eines ZFS-Speicher-Pools mit Datenspiegelung“ auf Seite 89.

Ein Speichergerät, das kein Protokollier- oder Cache-Gerät ist, kann nicht direkt aus einem Speicher-Pool mit Datenspiegelung entfernt werden. Dafür wurde ein RFE eingereicht.

Erstellen eines ZFS-Root-Pools

Sie können von einem ZFS-Root-Dateisystem installieren und booten. Beachten Sie die folgenden Root-Pool-Konfigurationsinformationen:

- Für das Root-Pool verwendete Datenträger müssen ein VTOC (SMI)-Label aufweisen, und der Pool muss mit Festplattenbereichen erstellt werden.

- Der Root-Pool muss als gespiegelte Konfiguration oder als einzelne Festplattenkonfiguration erstellt werden. Sie können mit dem Befehl `zpooladd` keine zusätzlichen Festplatten hinzufügen, um mehrere gespiegelte virtuelle Geräte der obersten Hierarchieebene zu erstellen, aber Sie können ein gespiegeltes virtuelles Gerät mit dem Befehl `zpool attach` erweitern.
- RAID-Z- oder Stripes-Konfigurationen werden nicht unterstützt.
- Der Root-Pool kann über kein separates Protokolliergerät verfügen.
- Bei dem Versuch, eine nicht unterstützte Pool-Konfiguration für ein Root-Pool zu verwenden, wird eine Meldung wie die folgende angezeigt:

```
ERROR: ZFS pool <pool-name> does not support boot environments
```

```
# zpool add -f rpool log c0t6d0s0
cannot add to 'rpool': root pool can not have multiple vdevs or separate logs
```

Weitere Informationen zum Installieren und Booten eines ZFS-Dateisystems finden Sie in Kapitel 5, „Installieren und Booten eines Oracle Solaris ZFS-Root-Dateisystems“.

Erstellen eines RAID-Z-Speicher-Pools

Das Erstellen eines RAID-Z-Pools mit einfacher Parität läuft ähnlich ab wie das Erstellen eines Pools mit Datenspiegelung. Der einzige Unterschied besteht darin, dass anstatt des Schlüsselworts `mirror` das Schlüsselwort `raidz` bzw. `raidz1` verwendet wird. Das folgende Beispiel zeigt die Erstellung eines Pools mit einem RAID-Z-Gerät, das aus fünf Datenträgern besteht:

```
# zpool create tank raidz c1t0d0 c2t0d0 c3t0d0 c4t0d0 /dev/dsk/c5t0d0
```

Dieses Beispiel zeigt, dass Datenträger angegeben werden können, indem ihre abgekürzten oder ihre vollständigen Gerätenamen verwendet werden. Sowohl `/dev/dsk/c5t0d0` als auch `c5t0d0` beziehen sich auf denselben Datenträger.

Eine RAID-Z-Konfiguration mit zwei- oder dreifacher Parität kann beim Erstellen eines Pools mithilfe des Schlüsselworts `raidz2` oder `raidsz3` angelegt werden. Beispiel :

```
# zpool create tank raidz2 c1t0d0 c2t0d0 c3t0d0 c4t0d0 c5t0d0
# zpool status -v tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          raidz2-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c3t0d0  ONLINE       0     0     0
            c4t0d0  ONLINE       0     0     0
            c5t0d0  ONLINE       0     0     0
```

```
errors: No known data errors

# zpool create tank raidz3 c0t0d0 c1t0d0 c2t0d0 c3t0d0 c4t0d0 c5t0d0 c6t0d0 c7t0d0
# zpool status -v tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          raidz3-0  ONLINE       0     0     0
            c0t0d0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c3t0d0  ONLINE       0     0     0
            c4t0d0  ONLINE       0     0     0
            c5t0d0  ONLINE       0     0     0
            c6t0d0  ONLINE       0     0     0
            c7t0d0  ONLINE       0     0     0

errors: No known data errors
```

Folgende Vorgänge werden in einer ZFS-RAID-Z-Konfiguration gegenwärtig unterstützt:

- Hinzufügen eines weiteren Festplattensatzes zu einer vorhandenen RAID-Z-Konfiguration, um eine neues virtuelles Gerät der obersten Hierarchieebene bereitzustellen. Weitere Informationen dazu finden Sie unter „Hinzufügen von Datenspeichergeräten zu einem Speicher-Pool“ auf Seite 83.
- Ersetzen einer oder mehrerer Festplatten in einer vorhandenen RAID-Z-Konfiguration, solange die Kapazität der neuen Festplatten größer oder gleich der Kapazität der zu ersetzenden Festplatten ist. Weitere Informationen dazu finden Sie unter „Austauschen von Geräten in einem Speicher-Pool“ auf Seite 95.

Folgende Vorgänge werden in einer RAID-Z-Konfiguration gegenwärtig *nicht* unterstützt:

- Hinzufügen einer weiteren Festplatte zu einer vorhandenen RAID-Z-Konfiguration.
- Entfernen einer Festplatte aus einer RAID-Z-Konfiguration, ausgenommen Entfernen einer Festplatte, die durch eine Ersatzfestplatte ersetzt wird.
- Ein Speichergerät, das kein Protokollier- oder Cache-Gerät ist, kann nicht direkt aus RAID-Z-Konfiguration entfernt werden. Dafür wurde ein RFE eingereicht.

Weitere Informationen zu RAID-Z-Konfigurationen finden Sie unter „Speicher-Pools mit RAID-Z-Konfiguration“ auf Seite 69.

Erstellen eines ZFS-Speicher-Pools mit Protokolliergeräten

Standardmäßig wird das ZIL aus Blöcken innerhalb des Haupt-Pools zugewiesen. Durch Verwendung getrennter Intent-Protokolliergeräte wie z. B. NVRAM oder eine speziell dafür

vorgesehene Festplatte kann jedoch eine höhere Leistung erreicht werden. Weitere Informationen zu ZFS-Protokolliergeräten finden Sie unter „Einrichten separater ZFS-Protokolliergeräte“ auf Seite 34.

Sie können ZFS-Protokolliergeräte während oder nach der Erstellung eines Speicher-Pools einrichten.

Das folgende Beispiel zeigt, wie ein Speicher-Pool mit Datenspiegelung und gespiegelten Protokolliergeräten erstellt wird:

```
# zpool create datap mirror c1t1d0 c1t2d0 mirror c1t3d0 c1t4d0 log mirror c1t5d0 c1t8d0
# zpool status datap
  pool: datap
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        datap       ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0
            c1t4d0  ONLINE       0     0     0
        logs
          mirror-2  ONLINE       0     0     0
            c1t5d0  ONLINE       0     0     0
            c1t8d0  ONLINE       0     0     0

errors: No known data errors
```

Informationen zum Wiederherstellen des Normalbetriebs nach dem Fehlschlag eines Protokolliergeräts finden Sie unter Beispiel 11–2.

Erstellen eines ZFS-Speicher-Pools mit Cache-Geräten

Sie können mit Cache-Geräten einen Pool erstellen, um Speicher-Pool-Daten im Cache zu speichern. Beispiel:

```
# zpool create tank mirror c2t0d0 c2t1d0 c2t3d0 cache c2t5d0 c2t8d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
        cache
```

```
          c2t5d0    ONLINE       0     0     0
          c2t8d0    ONLINE       0     0     0

errors: No known data errors
```

Beachten Sie Folgendes, wenn Sie die Erstellung eines ZFS-Speicher-Pools mit Cache-Geräten in Betracht ziehen:

- Cache-Geräte bieten die größte Leistungsverbesserung für die Direktspeicherung von Daten mit vorwiegend statischem Inhalt.
- Die Kapazität und Lesevorgänge können mithilfe des Befehls `zpool iostat` überwacht werden.
- Beim Erstellen eines Speicher-Pools können eines oder mehrere Cache-Geräte hinzugefügt werden. Die Geräte können auch nach der Erstellung des Pools hinzugefügt oder entfernt werden. Weitere Informationen finden Sie unter Beispiel 4–4.
- Cache-Geräte können nicht gespiegelt werden oder Teil einer RAID-Z-Konfiguration sein.
- Wenn auf einem Cache-Gerät ein Lesefehler entdeckt wird, wird der betreffende E/A-Lesevorgang wieder an das ursprüngliche Speicher-Pool-Gerät zurückgegeben, das Teil einer gespiegelten oder RAID-Z-Konfiguration sein kann. Der Inhalt der Cache-Geräte wird als flüchtig betrachtet, wie auch bei anderen Cache-Geräten des Systems.

Anzeigen von Informationen zu virtuellen Geräten in Storage-Pools

Jeder Speicher-Pool enthält eines oder mehrere virtuelle Geräte. Unter einem *virtuellen Gerät* versteht man eine interne Darstellung eines Speicher-Pools zur Beschreibung der Struktur der physischen Datenspeicherung und Fehlercharakteristika des Speicher-Pools. Somit repräsentiert ein virtuelles Gerät die Datenträger bzw. Dateien, die zum Erstellen eines Speicher-Pools verwendet werden. Ein Pool kann beliebig viele virtuelle Geräte auf oberster Konfigurationsebene aufweisen. Diese werden *vdev der obersten Ebene* genannt.

Wenn ein virtuelles Gerät der obersten Ebene zwei oder mehrere physische Geräte enthält, sorgt die Konfiguration für Datenredundanz durch Datenspiegelung oder virtuelle RAID-Z-Geräte. Diese virtuellen Geräte bestehen aus Datenträgern, Datenträgerbereichen oder Dateien. Als Spare werden spezielle virtuelle Geräte bezeichnet, die die verfügbaren Hot-Spares für einen Pool überwachen.

Das folgende Beispiel zeigt, wie ein Pool erstellt wird, der aus zwei virtuellen Geräten der obersten Ebene mit je zwei gespiegelten Festplatten besteht:

```
# zpool create tank mirror c1d0 c2d0 mirror c3d0 c4d0
```

Das folgende Beispiel zeigt, wie ein Pool erstellt wird, der aus einem virtuellen Gerät der obersten Ebene mit vier Festplatten besteht:

```
# zpool create mypool raidz2 c1d0 c2d0 c3d0 c4d0
```

Mit dem Befehl `zpool add` kann diesem Pool ein weiteres virtuelles Gerät der obersten Ebene hinzugefügt werden. Beispiel:

```
# zpool add mypool raidz2 c2d1 c3d1 c4d1 c5d1
```

Festplatten, Festplattenbereiche oder Dateien, die in nicht-redundanten Pools verwendet werden, fungieren als virtuelle Geräte der obersten Hierarchieebene. Speicher-Pools enthalten normalerweise mehrere virtuelle Geräte der obersten Hierarchieebene. ZFS verteilt Daten dynamisch mithilfe des sog. Stripe-Verfahrens über alle in der obersten Hierarchieebene befindlichen virtuellen Geräte eines Pools.

Virtuelle Geräte und die physischen Geräte, die in einem ZFS-Speicher-Pool enthalten sind, werden mit dem Befehl `zpool status` angezeigt. Beispiel:

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror-2  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
```

Behandlung von Fehlern beim Erstellen von ZFS-Speicher-Pools

Fehler beim Erstellen von Pools können in verschiedenen Situationen auftreten. Einige Fehlergründe sind sofort offensichtlich, wenn beispielsweise ein angegebenes Gerät nicht vorhanden ist, während andere Gründe nicht so einfach nachzuvollziehen sind.

Erkennen belegter Geräte

Vor dem Formatieren eines Datenspeichergeräts versucht ZFS herauszufinden, ob dieser Datenträger bereits von ZFS oder anderen Bereichen des Betriebssystems verwendet wird. Wenn der Datenträger bereits verwendet wird, sehen Sie in etwa folgende Fehlermeldung:

```
# zpool create tank c1t0d0 c1t1d0
invalid vdev specification
use '-f' to override the following errors:
/dev/dsk/c1t0d0s0 is currently mounted on /. Please see umount(1M).
/dev/dsk/c1t0d0s1 is currently mounted on swap. Please see swap(1M).
/dev/dsk/c1t1d0s0 is part of active ZFS pool zeepool. Please see zpool(1M).
```

Einige Fehler können mithilfe der Option `-f` ignoriert werden, die meisten Fehler jedoch nicht. Die folgenden Zustände können mithilfe der Option `-f` nicht ignoriert werden. Sie müssen diese Zustände manuell korrigieren:

Mounted file system (eingehängtes Dateisystem)
Der Datenträger oder einige seiner Bereiche enthalten ein gegenwärtig eingehängtes Dateisystem. Sie müssen diesen Fehler mit dem Befehl `umount` beheben.

File system in /etc/vfstab (Dateisystem in /etc/vfstab)
Der Datenträger enthält ein Dateisystem, das in der Datei `/etc/vfstab` aufgeführt, jedoch gegenwärtig nicht eingehängt ist. Zur Behebung dieses Fehlers müssen Sie die entsprechende Zeile aus der Datei `/etc/vfstab` entfernen bzw. diese Zeile auskommentieren.

Dedicated dump device (reserviertes Dump-Gerät)
Dieser Datenträger ist das reservierte Dump-Gerät für dieses System. Sie müssen diesen Fehler mit dem Befehl `dumpadm` beheben.

Part of a ZFS pool (Teil eines ZFS-Pools)
Der Datenträger bzw. die Datei gehört zu einem aktiven ZFS-Speicher-Pool. Sofern nicht mehr benötigt, entfernen Sie den anderen Pool mit dem Befehl `zpool destroy`. Oder verwenden Sie den Befehl `zpool detach`, um die Festplatte aus dem anderen Pool zu trennen. Sie können nur eine Festplatte aus einem Speicher-Pool mit Datenspiegelung trennen.

Die folgenden Überprüfungen auf Datenträgerbelegungen dienen als nützliche Warnungen und können mithilfe der Option `-f` manuell übergangen werden, um den Pool zu erstellen:

Contains a file system (Enthält ein Dateisystem)
Der Datenträger enthält ein bekanntes Dateisystem, obwohl es nicht eingehängt ist und nicht in Gebrauch zu sein scheint.

Part of volume (Teil eines Volumes)
Der Datenträger gehört zu einem Solaris Volume Manager-Volume.

Live Upgrade
Der Datenträger dient als alternative Boot-Umgebung für Oracle Solaris Live Upgrade.

Part of a exported ZFS pool (Teil eines exportierten ZFS-Pools)
Der Datenträger gehört zu einem Speicher-Pool, das exportiert bzw. manuell aus einem System entfernt wurde. Im letzteren Fall wird der Pool als `potenziell aktiv` gemeldet, da der Datenträger unter Umständen über das Netzwerk von einem anderen System belegt sein kann. Überprüfen Sie den Status vor beim Überschreiben eines potenziell aktiven

Datenträgers äußerst sorgfältig.

Das folgende Beispiel zeigt die Verwendung der Option `-f`:

```
# zpool create tank c1t0d0
invalid vdev specification
use '-f' to override the following errors:
/dev/dsk/c1t0d0s0 contains a ufs filesystem.
# zpool create -f tank c1t0d0
```

Im Normalfall sollten Sie Fehler beheben, anstatt sie mit Option `-f` zu ignorieren.

Inkongruente Replikationsmethoden

Die Erstellung von Pools mit unterschiedlichen Replikationsmethoden wird nicht empfohlen. Der Befehl `zpool` verhindert, dass Sie versehentlich einen Pool mit inkongruenten Redundanzebenen erstellen. Wenn Sie versuchen, einen Pool mit einer solchen Konfiguration zu erstellen, wird in etwa die folgende Fehlermeldung ausgegeben:

```
# zpool create tank c1t0d0 mirror c2t0d0 c3t0d0
invalid vdev specification
use '-f' to override the following errors:
mismatched replication level: both disk and mirror vdevs are present
# zpool create tank mirror c1t0d0 c2t0d0 mirror c3t0d0 c4t0d0 c5t0d0
invalid vdev specification
use '-f' to override the following errors:
mismatched replication level: 2-way mirror and 3-way mirror vdevs are present
```

Sie können diese Fehler mit der Option `-f` ignorieren, obwohl dies nicht empfohlen wird. Der Befehl warnt Sie darüber hinaus auch, wenn Sie versuchen, einen Pool mit Datenspiegelung bzw. RAID-Z zu erstellen, das Datenträger unterschiedlicher Kapazität verwendet. Obwohl eine solche Konfiguration zulässig ist, führen inkongruente Redundanzebenen dazu, dass die Kapazität größerer Datenträger ungenutzt bleibt. Die Option `-f` wird benötigt, um die Warnung zu ignorieren.

Ausführen eines Testlaufs für die Erstellung eines Speicher-Pools

Versuche, einen Pool zu erstellen, können unerwartet und aus verschiedenen Gründen fehlschlagen, und das Formatieren von Datenträgern ist ein potentiell schädlicher Vorgang. Aus diesen Gründen verfügt der Befehl `zpool create` über eine zusätzliche Option (`-n`), mit der das Erstellen eines Pools simuliert wird, ohne das Gerät tatsächlich zu beschreiben. Mit der Option *dry run* wird die Belegung von Datenträgern sowie die Evaluierung der Replikationsmethoden überprüft, und es werden im Verlauf dieses Vorgangs eventuell auftretende Fehler gemeldet. Wenn keine Fehler gefunden wurden, wird in etwa die folgende Meldung ausgegeben:

```
# zpool create -n tank mirror c1t0d0 c1t1d0
would create 'tank' with the following layout:
```

```
tank
  mirror
    c1t0d0
    c1t1d0
```

Einige Fehler werden jedoch erst angezeigt, wenn der Pool tatsächlich erstellt wird. Ein Beispiel für einen solchen häufig auftretenden Fehler ist die mehrmalige Angabe eines Datenträgers in der gleichen Konfiguration. Dieser Fehler kann erst beim tatsächlichen Schreiben von Daten zuverlässig erkannt werden. Aus diesem Grunde kann es sein, dass mit dem Befehl `zpool create -n` eine erfolgreiche Ausführung gemeldet wird, aber der Pool nicht erstellt werden kann, wenn der Befehl ohne diese Option ausgeführt wird.

Standard-Einhängepunkt für Speicher-Pools

Bei der Erstellung eines Speicher-Pools ist /*pool-name* der Standard-Einhängepunkt für das Dataset der obersten Hierarchieebene. Dieses Verzeichnis darf noch nicht vorhanden oder muss leer sein. Falls das Verzeichnis noch nicht existiert, wird es autoamtisch erstellt. Wenn das Verzeichnis leer ist, wird das Root-Dataset in dieses vorhandene Verzeichnis eingehängt. Mithilfe der Option `-m` des Befehls `zpool create` können Sie Pools mit anderen Standard-Einhängepunkten erstellen. Beispiel:

```
# zpool create home c1t0d0
default mountpoint '/home' exists and is not empty
use '-m' option to provide a different default
# zpool create -m /export/zfs home c1t0d0
```

Mit diesem Befehl wird der neue Pool `home` sowie das Dataset `home` mit dem Einhängepunkt `/export/zfs` erstellt.

Weitere Informationen zu Einhängepunkten finden Sie unter „Verwalten von ZFS-Einhängepunkten" auf Seite 211.

Löschen von ZFS-Speicher-Pools

Pools werden mit dem Befehl `zpool destroy` gelöscht. Dieser Befehl löscht einen Pool auch dann, wenn er eingehängte Datasets enthält.

```
# zpool destroy tank
```



Achtung – Seien Sie beim Löschen von Pools äußerst vorsichtig. Stellen Sie sicher, dass der richtige Pool gelöscht wird und Sie Sicherungskopien der Daten dieses Pools erstellt haben. Falls versehentlich der falsche Pool gelöscht wurde, kann er eventuell wiederhergestellt werden. Weitere Informationen dazu finden Sie unter „Wiederherstellen gelöschter ZFS-Speicher-Pools" auf Seite 120.

Löschen eines Pools mit fehlerhaften Geräten

Wenn ein Pool gelöscht wird, müssen Daten auf den Datenträger geschrieben werden, um anzuzeigen, dass der Pool nicht mehr verfügbar ist. Diese Statusinformationen verhindern, dass beim Durchführen von Imports diese Datenspeichergeräte als potenziell verfügbar angezeigt werden. Auch wenn Datenspeichergeräte nicht verfügbar sind, kann der Pool gelöscht werden. Die erforderlichen Statusinformationen werden in diesem Fall jedoch nicht auf den nicht verfügbaren Datenträgern gespeichert.

Nach einer entsprechenden Reparatur werden diese Datenspeichergeräte bei der Erstellung neuer Pools als *potenziell aktiv* gemeldet. Sie werden als verfügbare Geräte angezeigt, wenn Sie nach Pools suchen, die importiert werden sollen. Wenn ein Pool so viele fehlerhafte Datenspeichergeräte enthält, dass der Pool selbst als fehlerhaft erkannt wird (d.h. das Gerät der obersten Hierarchieebene ist fehlerhaft), gibt der Befehl eine Warnmeldung aus und kann ohne die Option `-f` nicht fortgesetzt werden. Diese Option ist erforderlich, da der Pool nicht geöffnet werden kann und somit nicht bekannt ist, ob dort Daten gespeichert sind. Beispiel:

```
# zpool destroy tank
cannot destroy 'tank': pool is faulted
use '-f' to force destruction anyway
# zpool destroy -f tank
```

Weitere Informationen zum Funktionsstaus von Pools und Datenspeichergeräten finden Sie unter „Ermitteln des Funktionsstatus von ZFS-Speicher-Pools" auf Seite 112.

Weitere Informationen zum Importieren von Pools finden Sie unter „Importieren von ZFS-Speicher-Pools" auf Seite 119.

Verwalten von Datenspeichergeräten in ZFS-Speicher-Pools

Die meisten grundlegenden Informationen zu Datenspeichergeräten sind in „Komponenten eines ZFS-Speicher-Pools" auf Seite 65 enthalten. Nach dem Erstellen eines Pools können Sie zum Verwalten der zum Pool gehörenden Datenspeichergeräte verschiedene Aufgaben ausführen.

Hinzufügen von Datenspeichergeräten zu einem Speicher-Pool

Durch Hinzufügen eines neuen Gerätes der obersten Hierarchieebene können Sie einen Pool dynamisch um Festplattenkapazität erweitern. Die zusätzliche Festplattenkapazität steht allen im Pool enthaltenen Datasets sofort zur Verfügung. Mit dem Befehl `zpool add` fügen Sie einem Pool ein neues virtuelles Gerät hinzu. Beispiel:

```
# zpool add zeepool mirror c2t1d0 c2t2d0
```

Das Format zum Angeben dieser virtuellen Geräte entspricht dem Format für den Befehl `zpool create`. Es wird geprüft, ob die betreffenden Datenspeichergeräte belegt sind, und der Befehl kann die Redundanzebene ohne die Option `-f` nicht ändern. Der Befehl unterstützt auch die Option `-n` zum Durchführen eines Testlaufs. Beispiel:

```
# zpool add -n zeepool mirror c3t1d0 c3t2d0
would update 'zeepool' to the following configuration:
      zeepool
        mirror
            c1t0d0
            c1t1d0
        mirror
            c2t1d0
            c2t2d0
        mirror
            c3t1d0
            c3t2d0
```

Mit dieser Befehlsyntax werden die gespiegelten Geräte `c3t1d0` und `c3t2d0` zur vorhandenen Konfiguration des Pools`zeepool` hinzugefügt.

Weitere Informationen zur Überprüfung von virtuellen Geräten finden Sie unter „Erkennen belegter Geräte“ auf Seite 78.

BEISPIEL 4–1 Hinzufügen von Festplatten in eine ZFS-Konfiguration mit Datenspiegelung

Im folgenden Beispiel wird eine ZFS-Konfiguration mit Datenspiegelung auf einem Oracle Sun Fire x4500-System um eine weitere Spiegelplatte ergänzt.

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
```

BEISPIEL 4–1 Hinzufügen von Festplatten in eine ZFS-Konfiguration mit Datenspiegelung *(Fortsetzung)*

```
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0

errors: No known data errors
# zpool add tank mirror c0t3d0 c1t3d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror-2  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
```

BEISPIEL 4–2 Hinzufügen von Datenträgern zu einer RAID-Z-Konfiguration

Datenträger können in ähnlicher Weise auch zu einer RAID-Z-Konfiguration hinzugefügt werden. Das folgende Beispiel zeigt, wie ein Speicher-Pool mit einem aus drei Festplatten bestehenden RAID-Z-Gerät in ein Speicher-Pool mit zwei aus je drei Datenträgern bestehenden RAID-Z-Geräten umgewandelt werden kann.

```
# zpool status rzpool
  pool: rzpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        rzpool      ONLINE       0     0     0
          raidz1-0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0
            c1t4d0  ONLINE       0     0     0

errors: No known data errors
# zpool add rzpool raidz c2t2d0 c2t3d0 c2t4d0
# zpool status rzpool
  pool: rzpool
 state: ONLINE
 scrub: none requested
config:
```

BEISPIEL 4–2 Hinzufügen von Datenträgern zu einer RAID-Z-Konfiguration *(Fortsetzung)*

```
        NAME        STATE     READ WRITE CKSUM
        rzpool      ONLINE       0     0     0
          raidz1-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0
          raidz1-1  ONLINE       0     0     0
            c2t2d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
            c2t4d0  ONLINE       0     0     0

errors: No known data errors
```

BEISPIEL 4–3 Hinzufügen und Entfernen eines gespiegelten Protokolliergeräts

Das folgende Beispiel zeigt, wie einem Speicher-Pool mit Datenspiegelung ein gespiegeltes Protokolliergerät hinzugefügt werden kann.Weitere Informationen zum Einsatz von Protokolliergeräten in Speicher-Pools finden Sie unter „Einrichten separater ZFS-Protokolliergeräte“ auf Seite 34.

```
# zpool status newpool
  pool: newpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        newpool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
            c0t5d0  ONLINE       0     0     0

errors: No known data errors
# zpool add newpool log mirror c0t6d0 c0t7d0
# zpool status newpool
  pool: newpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        newpool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
            c0t5d0  ONLINE       0     0     0
        logs
          mirror-1  ONLINE       0     0     0
            c0t6d0  ONLINE       0     0     0
            c0t7d0  ONLINE       0     0     0

errors: No known data errors
```

BEISPIEL 4–3 Hinzufügen und Entfernen eines gespiegelten Protokolliergeräts *(Fortsetzung)*

Sie können zur Datenspiegelung an ein vorhandenes Protokolliergerät ein weiteres Protokolliergerät anschließen. Dies entspricht dem Verbinden eines Speichergeräts in einem Speicher-Pool ohne Datenspiegelung.

Protokolliergeräte können mithilfe des Befehls `zpool remove` entfernt werden. Das gespiegelte Protokolliergerät, um das es im vorherigen Beispiel geht, kann durch Angabe des Arguments `mirror-1` entfernt werden. Beispiel:

```
# zpool remove newpool mirror-1
# zpool status newpool
  pool: newpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        newpool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
            c0t5d0  ONLINE       0     0     0

errors: No known data errors
```

Wenn Ihre Poolkonfiguration nur ein Protokolliergerät enthält, können Sie das Gerät entfernen, indem Sie den Namen des Geräts angeben. Beispiel:

```
# zpool status pool
  pool: pool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        ONLINE       0     0     0
          raidz1-0  ONLINE       0     0     0
            c0t8d0  ONLINE       0     0     0
            c0t9d0  ONLINE       0     0     0
        logs
          c0t10d0   ONLINE       0     0     0

errors: No known data errors
# zpool remove pool c0t10d0
```

BEISPIEL 4–4 Hinzufügen und Entfernen von Cache-Geräten

Sie können Cache-Geräte zu Ihrem ZFS-Speicher-Pool hinzufügen oder Cache-Geräte aus Ihrem ZFS-Speicher-Pool entfernen, wenn diese nicht benötigt werde.

Verwenden Sie den Befehl `zpool add` zum Hinzufügen von Cache-Geräten. Beispiele:

```
# zpool add tank cache c2t5d0 c2t8d0
# zpool status tank
```

BEISPIEL 4–4 Hinzufügen und Entfernen von Cache-Geräten *(Fortsetzung)*

```
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
        cache
          c2t5d0    ONLINE       0     0     0
          c2t8d0    ONLINE       0     0     0

errors: No known data errors
```

Cache-Geräte können nicht gespiegelt werden oder Teil einer RAID-Z-Konfiguration sein.

Verwenden Sie den Befehl `zpool remove` zum Entfernen von Cache-Geräten. Beispiel:

```
# zpool remove tank c2t5d0 c2t8d0
# zpool status tank
 pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0

errors: No known data errors
```

Der Befehl `zpool remove` unterstützt derzeit nur das Entfernen von Hot-Spares, Protokolliergeräten und Cache-Geräten. Geräte, die zur primären gespiegelten Pool-Konfiguration gehören, können mit dem Befehl `zpool detach` entfernt werden. Geräte ohne Redundanz und RAID-Z-Geräte können nicht aus einem Pool entfernt werden.

Weitere Informationen zur Verwendung von Cache-Geräten in einem ZFS-Speicher-Pool finden Sie unter „Erstellen eines ZFS-Speicher-Pools mit Cache-Geräten“ auf Seite 76.

Verbinden und Trennen von Geräten in einem Speicher-Pool

Zusätzlich zum Befehl `zpool add` können Sie mit dem Befehl `zpool attach` zu einem Gerät mit oder ohne Datenspiegelung ein neues Datenspeichergerät hinzufügen.

Wenn Sie eine Festplatte einbinden möchten, um einen gespiegelten Root-Pool zu erstellen, gehen Sie wie unter „So erstellen Sie einen gespiegelten Root-Pool (nach der Installation)“ auf Seite 136 beschrieben vor.

Wenn Sie eine Festplatte in einem ZFS-Root-Pool ersetzen möchten, gehen Sie wie unter „So ersetzen Sie eine Festplatte im ZFS-Root-Pool“ auf Seite 177 beschrieben vor.

BEISPIEL 4–5 Umwandlung eines Speicher-Pools mit zweifacher Datenspiegelung in einen Speicher-Pool mit dreifacher Datenspiegelung

In diesem Beispiel ist `zeepool` eine vorhandene zweifache Datenspiegelungskonfiguration, die durch Verbinden des Geräts `c2t1d0` mit dem vorhandenen Gerät `c1t1d0` in eine dreifache Datenspiegelungskonfiguration umgewandelt wird.

```
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
# zpool attach zeepool c1t1d0 c2t1d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Fri Jan  8 12:59:20 2010
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0  592K resilvered

errors: No known data errors
```

Wenn das betreffende Gerät zu einer dreifachen Datenspiegelungskonfiguration gehört, wird durch Verbinden des neuen Geräts eine vierfache Datenspiegelungskonfiguration erstellt usw. In jedem Fall wird sofort das Resilvering durchgeführt (d. h., die gespiegelten Daten werden auf das neue Gerät in der Datenspiegelungskonfiguration übertragen).

BEISPIEL 4–6 Umwandeln eines ZFS-Speicher-Pools ohne Redundanz in einen ZFS-Speicher-Pool mit Datenspiegelung

Außerdem können Sie mithilfe des Befehls `zpool attach` einen Speicher-Pool ohne Redundanz in einen Pool mit Redundanz umwandeln. Beispiel:

BEISPIEL 4–6 Umwandeln eines ZFS-Speicher-Pools ohne Redundanz in einen ZFS-Speicher-Pool mit Datenspiegelung *(Fortsetzung)*

```
# zpool create tank c0t1d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:
        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          c0t1d0    ONLINE       0     0     0

errors: No known data errors
# zpool attach tank c0t1d0 c1t1d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Fri Jan  8 14:28:23 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0  73.5K resilvered

errors: No known data errors
```

Mit dem Befehl `zpool detach` kann ein Datenspeichergerät aus einem Pool mit Datenspiegelungskonfiguration abgetrennt werden. Beispiel:

```
# zpool detach zeepool c2t1d0
```

Dieser Vorgang ist allerdings nicht ausführbar, wenn keine anderen Replikationen der betreffenden Daten vorhanden sind. Beispiel:

```
# zpool detach newpool c1t2d0
cannot detach c1t2d0: only applicable to mirror and replacing vdevs
```

Erstellen eines neuen Pools durch Teilen eines ZFS-Speicher-Pools mit Datenspiegelung

Eine ZFS-Speicher-Pool mit Datenspiegelung kann mithilfe des Befehls `zpool split` schnell als Sicherungs-Pool geklont werden.

Diese Funktion kann derzeit nicht zum Teilen eines Root-Pools mit Datenspiegelung verwendet werden.

Mithilfe des Befehls `zpool split` können Sie Festplatten von einem ZFS-Speicher-Pool mit Datenspiegelung trennen, um einen neuen Pool mit einer der getrennten Festplatten zu erstellen. Der Inhalt des neuen Pools ist mit dem des ursprünglichen ZFS-Speicher-Pools mit Datenspiegelung identisch.

Durch den Befehl `zpool split` wird die letzte Festplatte eines Pools mit Datenspiegelung standardmäßig getrennt und für den neuen Pool verwendet. Nach der Teilung importieren Sie den neuen Pool. Beispiel:

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0

errors: No known data errors
# zpool split tank tank2
# zpool import tank2
# zpool status tank tank2
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          c1t0d0    ONLINE       0     0     0

errors: No known data errors

  pool: tank2
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank2       ONLINE       0     0     0
          c1t2d0    ONLINE       0     0     0

errors: No known data errors
```

Sie können feststellen, welche Festplatte für den neuen Pool verwendet werden soll, indem Sie sie mithilfe des Befehls `zpool split` angeben. Beispiel:

```
# zpool split tank tank2 c1t0d0
```

Vor der tatsächlichen Teilung werden die im Speicher enthaltenen Daten auf die gespiegelten Festplatten ausgespeichert. Nach dem Ausspeichern wird die Festplatte vom Pool getrennt und

erhält eine neue Pool-GUID. Die neue Pool-GUID wird generiert, damit der Pool auf dasselbe System importiert werden kann, auf dem er geteilt wurde.

Wenn der Pool, der geteilt werden soll, keine standardmäßigen Dataset-Einhängepunkte hat und der neue Pool auf demselben System erstellt wird, müssen Sie die Option `zpool split -R` verwenden, um ein alternatives Root-Verzeichnis für den neuen Pool zu bestimmen, damit es nicht zu Konflikten zwischen vorhandenen Einhängepunkten kommt. Beispiel:

```
# zpool split -R /tank2 tank tank2
```

Wenn Sie nicht die Option `zpool split -R` verwenden und feststellen, dass Einhängepunkte in Konflikt geraten, sobald Sie versuchen, den neuen Pool zu importieren, importieren Sie den neuen Pool mithilfe der Option `-R`. Wenn der neue Pool auf einem anderen System erstellt wird, ist die Angabe eines alternativen Root-Verzeichnisses nicht nötig, es sei denn, es treten Einhängepunkt-Konflikte auf.

Berücksichtigen Sie Folgendes, bevor Sie die Funktion `zpool split` verwenden:

- Diese Funktion steht für RAIDZ-Konfigurationen oder nicht redundante Pools mit mehreren Festplatten nicht zur Verfügung.
- Datenverarbeitungs- und Anwendungsprozesse sollten unterbrochen werden, bevor die Funktion `zpool split` verwendet wird.
- Es ist wichtig, dass der Ausspeicherungsbefehl von den Festplatten akzeptiert und nicht ignoriert wird.
- Ein Pool kann nicht geteilt werden, wenn das Resilvering im Gange ist.
- Ein Pool mit Datenspiegelung kann optimal geteilt werden, wenn dieser zwei bis drei Festplatten enthält und die letzte Festplatte des Pools für den neuen Pool verwendet wird. Anschließend können Sie den Befehl `zpool attach` verwenden, um den ursprünglichen Speicher-Pool mit Datenspiegelung wiederherzustellen oder den neuen Pool in einem Speicher-Pool mit Datenspiegelung umzuwandeln. Derzeit besteht keine Möglichkeit, mithilfe dieser Funktion einen *neuen* Pool mit Datenspiegelung aus einem *vorhandenen* Pool mit Datenspiegelung zu erstellen..
- Wenn der vorhandene Pool ein Pool mit dreifacher Datenspiegelung ist, enthält der neue Pool nach der Teilung eine Festplatte. Wenn der vorhandene Pool ein Pool mit zweifacher Datenspiegelung ist und zwei Festplatten enthält, entstehen durch die Teilung zwei nicht redundante Pools mit zwei Festplatten. Sie müssen zwei weitere Festplatten einbinden, um die nicht redundanten Pools in Pools mit Datenspiegelung umzuwandeln.
- Ein gute Methode zur Beibehaltung der Daten während einer Teilung ist es, einen Speicher-Pool mit Datenspiegelung, der drei Festplatten enthält, zu teilen, sodass der ursprüngliche Pool nach der Teilung zwei Festplatten mit Datenspiegelung enthält.

BEISPIEL 4–7 Teilung eines ZFS-Speicher-Pools mit Datenspiegelung

Im folgenden Beispiel wird ein Speicher-Pool mit Datenspiegelung namens `trinity`, der drei Festplatten enthält (`c1t0d0`, `c1t2d0` und `c1t3d0`), geteilt. Dadurch entstehen der Pool mit

BEISPIEL 4–7 Teilung eines ZFS-Speicher-Pools mit Datenspiegelung *(Fortsetzung)*

Datenspiegelung `trinity`, der die Festplatten `c1t0d0` und `c1t2d0` enthält, und der neue Pool `neo`, der die Festplatte `c1t3d0` enthält. Der Inhalt beider Pools ist identisch.

```
# zpool status trinity
  pool: trinity
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        trinity     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
# zpool split trinity neo
# zpool import neo
# zpool status trinity neo
  pool: neo
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        neo         ONLINE       0     0     0
          c1t3d0    ONLINE       0     0     0

errors: No known data errors

  pool: trinity
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        trinity     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0

errors: No known data errors
```

In- und Außerbetriebnehmen von Geräten in einem Speicher-Pool

Einzelne Datenspeichergeräte können in ZFS in und außer Betrieb genommen werden. Wenn Hardwarekomponenten unzuverlässig sind und nicht richtig funktionieren, schreibt ZFS trotzdem weiter Daten auf das betreffende Gerät bzw. liest sie von diesem und geht davon aus, dass dieser Zustand nicht von Dauer ist. Wenn dieser Zustand andauert, können Sie ZFS

anweisen, das Gerät zu ignorieren, wodurch es außer Betrieb genommen wird. ZFS sendet dann keine Anforderungen an das außer Betrieb genommene Gerät.

Hinweis – Zum Austauschen von Datenspeichergeräten müssen diese vorher nicht außer Betrieb genommen werden.

Mit dem Befehl `zpool offline` können Sie Datenspeichergeräte zeitweilig außer Betrieb nehmen. Wenn Sie beispielsweise ein Array aus einer Gruppe Fibre Channel-Switches herausnehmen und an eine andere Gruppe anschließen müssen, können Sie die LU-Nummern aus dem Array, das in den ZFS-Speicher-Pools verwendet wird, deaktivieren. Nachdem Sie das Array neu angeschlossen haben und das Array in der neuen Gruppe der Switches einsatzbereit ist, können Sie die LU-Nummern wieder aktivieren. Daten, die zu den Speicher-Pools hinzugefügt wurden, als die LU-Nummern deaktiviert waren, werden nach der erneuten Aktivierung der LU-Nummern durch Resilvering auf diese übertragen.

Diese Situation ist möglich, da angenommen wird, dass die Datenträger nach dem Anschließen an die neuen Switches von den Systemen erkannt werden. Dies kann unter Umständen über andere Controller als zuvor geschehen, und die betreffenden Pools sind mit Datenspiegelung oder RAID-Z konfiguriert.

Außerbetriebnehmen eines Geräts

Datenspeichergeräte werden mit dem Befehl `zpool offline` außer Betrieb genommen. Wenn es sich bei dem Datenspeichergerät um eine Festplatte handelt, kann es mit dem vollständigen Pfad oder einer Kurzbezeichnung angegeben werden. Beispiel:

```
# zpool offline tank c1t0d0
bringing device c1t0d0 offline
```

Beim Außerbetriebnehmen von Datenspeichergeräten sollten Sie Folgendes berücksichtigen:

- Pools können nicht außer Betrieb genommen werden, wenn dadurch die Datensicherheit beeinträchtigt wird. So können Sie beispielsweise zwei Geräte nicht aus einer `raidz1`-Konfiguration oder kein virtuelles Gerät der obersten Hierarchieebene außer Betrieb nehmen.

  ```
  # zpool offline tank c1t0d0
  cannot offline c1t0d0: no valid replicas
  ```
- Der `OFFLINE`-Status ist dauerhaft, d. h. das Gerät bleibt auch bei einem Systemneustart außer Betrieb.

  Mit dem Befehl `zpool offline -t` können Sie ein Datenspeichergerät zeiweilig außer Betrieb nehmen. Beispiel:

  ```
  # zpool offline -t tank c1t0d0
   bringing device 'c1t0d0' offline
  ```

Bei einem Systemneustart wird dieses Gerät dann automatisch in den `ONLINE`-Status gebracht.

- Wenn ein Gerät außer Betrieb genommen wurde, wird es nicht automatisch vom Speicher-Pool getrennt. Wenn Sie versuchen, das betreffende Gerät für einen anderen Pool zu verwenden (selbst wenn dieser Pool gelöscht wurde), wird eine Meldung wie die folgende angezeigt:

  device `is part of exported or potentially active ZFS` *pool*`. Please see zpool(1M)`

  Wenn Sie das außer Betrieb genommene Gerät für einen anderen Pool verwenden möchten, nachdem der ursprüngliche Pool gelöscht wurde, müssen Sie zunächst das Gerät wieder in Betrieb nehmen und dann den ursprünglichen Pool löschen.

  Eine andere Methode, ein Gerät eines anderen Speicher-Pools zu nutzen, während der ursprüngliche Pool beibehalten wird, besteht darin, das vorhandene Gerät im ursprünglichen Pool durch ein anderes, vergleichbares Gerät zu ersetzen. Weitere Informationen zum Austauschen von Geräten finden Sie unter „Austauschen von Geräten in einem Speicher-Pool“ auf Seite 95.

Wenn Sie den Pool-Status abfragen, weisen außer Betrieb genommene Geräte den Status `OFFLINE` auf. Weitere Informationen zum Abfragen des Pool-Status finden Sie unter „Abfragen des Status von ZFS-Speicher-Pools“ auf Seite 106.

Weitere Informationen zum Funktionsstatus von Datenspeichergeräten finden Sie unter „Ermitteln des Funktionsstatus von ZFS-Speicher-Pools“ auf Seite 112.

Inbetriebnehmen eines Gerätes

Nach dem Außerbetriebnehmen eines Geräts kann es mit dem Befehl `zpool online` wieder in Betrieb genommen werden. Beispiel:

```
# zpool online tank c1t0d0
bringing device c1t0d0 online
```

Nach dem Inbetriebnehmen des Gerätes werden alle Daten, die im Pool gespeichert wurden, mit dem neu verfügbaren Gerät synchronisiert. Bitte beachten Sie, dass Sie eine Festplatte nicht ersetzen können, indem Sie ein Gerät in Betrieb nehmen. Wenn Sie ein Gerät außer Betrieb nehmen, es austauschen und versuchen, es wieder in Betrieb zu nehmen, verursacht das einen Fehlerzustand.

Wenn Sie versuchen, ein fehlerhaftes Gerät in Betrieb zu nehmen, wird eine Meldung wie die folgende angezeigt:

```
# zpool online tank c1t0d0
warning: device 'c1t0d0' onlined, but remains in faulted state
use 'zpool replace' to replace devices that are no longer present
```

Die Fehlermeldung kann auch an der Konsole angezeigt oder in die Datei unter `/var/adm/messages` geschrieben werden. Beispiel:

```
SUNW-MSG-ID: ZFS-8000-D3, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Wed Jun 30 14:53:39 MDT 2010
PLATFORM: SUNW,Sun-Fire-880, CSN: -, HOSTNAME: neo
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: 504a1188-b270-4ab0-af4e-8a77680576b8
DESC: A ZFS device failed.  Refer to http://sun.com/msg/ZFS-8000-D3 for more information.
AUTO-RESPONSE: No automated response will occur.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Run 'zpool status -x' and replace the bad device.
```

Weitere Informationen zum Austauschen fehlerhafter Geräte finden Sie unter „Abhilfe bei Nichtverfügbarkeit eines Geräts“ auf Seite 309.

Mithilfe des Befehls `zpool online -e` können Sie eine LU-Nummer erweitern. Eine LU-Nummer, die zu einem Pool hinzugefügt wird, wird standardmäßig nicht auf ihre volle Größe erweitert, wenn die Pool-Eigenschaft `autoexpand` nicht aktiviert ist. Mithilfe des Befehls `zpool online -e` können Sie die LU-Nummer automatisch erweitern, selbst dann, wenn die LU-Nummer bereits aktiviert wurde oder gerade deaktiviert ist. Beispiel:

```
# zpool online -e tank c1t13d0
```

Löschen von Gerätefehlern im Speicher-Pool

Wenn ein Datenspeichergerät aufgrund von Fehlern, die in der Ausgabe von `zpool status` aufgeführt werden, außer Betrieb genommen wird, können Sie diese Fehler mithilfe des Befehls `zpool clear` löschen.

Wenn keine Argumente angegeben werden, löscht der Befehl alle Gerätefehler eines Pools. Beispiel:

```
# zpool clear tank
```

Wenn ein Gerät oder mehrere Geräte angegeben werden, löscht der Befehl nur die zu den betreffenden Geräten gehörenden Fehler. Beispiel:

```
# zpool clear tank c1t0d0
```

Weitere Informationen zum Löschen von `zpool`-Fehlern finden Sie unter „Löschen vorübergehender Fehler“ auf Seite 313.

Austauschen von Geräten in einem Speicher-Pool

Mit dem Befehl `zpool replace` können Sie Datenspeichergeräte in einem Speicher-Pool austauschen.

Wenn Sie in einem Pool mit Redundanz ein Datenspeichergerät an der gleichen Stelle durch ein anderes Datenspeichergerät ersetzen, brauchen Sie nur das ersetzte Datenspeichergerät

anzugeben. ZFS erkennt, dass das Gerät eine andere Festplatte ist, die sich an derselben Stelle auf einer Hardwarekomponente befindet. Wenn Sie beispielsweise eine ausgefallene Festplatte (c1t1d0) durch Auswechseln an der gleichen Stelle ersetzen wollen, verwenden Sie folgende Syntax:

```
# zpool replace tank c1t1d0
```

Wenn Sie eine Gerät in einem Speicher-Pool mit einer Festplatte an einer anderen physischen Stelle ersetzen wollen, müssen Sie beide Geräte angeben. Beispiel:

```
# zpool replace tank c1t1d0 c1t2d0
```

Wenn Sie eine Festplatte in einem ZFS-Root-Pool ersetzen, lesen Sie unter „So ersetzen Sie eine Festplatte im ZFS-Root-Pool" auf Seite 177 nach.

Es folgen die grundlegenden Schritte zum Austauschen von Datenträgern:

- Nehmen Sie den Datenträger wenn nötig mit dem Befehl zpool offline außer Betrieb.
- Bauen Sie die zu erstetzende Festplatte aus.
- Setzen Sie die Ersatzfestplatte ein.
- Führen Sie den Befehl zpool replace aus. Beispiel:

  ```
  # zpool replace tank c1t1d0
  ```
- Nehmen Sie den Datenträger mit dem Befehl zpool online in Betrieb.

Bei manchen Systemen, wie etwa bei Sun Fire x4500, muss eine Festplatte vor der Außerbetriebnahme dekonfiguriert werden. Wenn Sie bei diesem System eine Festplatte an ein und demselben Steckplatz austauschen, genügt es, den Befehl zpool replace wie im ersten Beispiel dieses Abschnitt beschrieben auszuführen.

Ein Beispiel für das Austauschen eines Datenträgers auf einem Sun Fire X4500-System finden Sie in Beispiel 11–1.

Beachten Sie beim Auswechseln von Datenspeichergeräten in einem ZFS-Speicher-Pool Folgendes:

- Wenn Sie die Eigenschaft autoreplace des Pools auf on setzen, wird jedes neue Datenspeichergerät, das sich an der physischen Stelle eines zuvor zum Pool gehörenden Datenspeichergeräts befindet, automatisch formatiert und ersetzt. Der Befehl zpool replace muss nicht verwendet werden, wenn diese Eigenschaft aktiviert ist. Dieses Leistungsmerkmal ist möglicherweise nicht auf jeder Art von Hardware verfügbar.
- Die Kapazität des Austauschgeräts muss der Kapazität der kleinsten Festplatte in einer Datenspiegelungs- bzw. RAID-Z-Konfiguration entsprechen oder größer sein.
- Wenn ein Austauschgerät – dessen Kapazität größer ist als die des Geräts, das ausgetauscht wird – zu einem Pool hinzugefügt wird, wird es nicht automatisch auf seine volle Kapazität erweitert. Der Pool-Eigenschaftswert autoexpand bestimmt, ob eine

Austausch-LU-Nummer auf ihre volle Größe erweitert wird, wenn die Festplatte zum Pool hinzugefügt wird. Standardmäßig ist die Eigenschaft `autoexpand` aktiviert. Sie können diese Eigenschaft aktivieren, um die LU-Nummer zu erweitern, bevor oder nachdem die größere LU-Nummer zum Pool hinzugefügt wird.

Im folgenden Beispiel werden zwei 16-GB-Festplatten in einem Pool mit Datenspiegelung durch zwei 72-GB-Festplatten ersetzt. Nach dem Ersetzen der Festplatten wird die Eigenschaft `autoexpand` aktiviert, um die LU-Nummern auf die volle Größe zu erweitern.

```
# zpool create pool mirror c1t16d0 c1t17d0
# zpool status
  pool: pool
 state: ONLINE
 scrub: none requested
config:

        NAME         STATE     READ WRITE CKSUM
        pool         ONLINE       0     0     0
          mirror     ONLINE       0     0     0
            c1t16d0  ONLINE       0     0     0
            c1t17d0  ONLINE       0     0     0

zpool list pool
NAME   SIZE   ALLOC  FREE    CAP  HEALTH  ALTROOT
pool  16.8G  76.5K  16.7G     0%  ONLINE  -
# zpool replace pool c1t16d0 c1t1d0
# zpool replace pool c1t17d0 c1t2d0
# zpool list pool
NAME   SIZE   ALLOC  FREE    CAP  HEALTH  ALTROOT
pool  16.8G  88.5K  16.7G     0%  ONLINE  -
# zpool set autoexpand=on pool
# zpool list pool
NAME   SIZE   ALLOC  FREE    CAP  HEALTH  ALTROOT
pool  68.2G   117K  68.2G     0%  ONLINE  -
```

- Das Austauschen mehrerer Festplatten in einem großen Pool ist zeitaufwändig, da die Daten mithilfe von Resilvering auf die neuen Festplatten aufgespielt werden müssen. Außerdem sollten Sie zwischen dem Austausch von Festplatten den Befehl `zpool scrub` ausführen, um sicherzustellen, dass die Austauschgeräte ordnungsgemäß funktionieren und Daten fehlerfrei geschrieben werden.
- Wenn eine ausgefallene Festplatte automatisch durch eine Hot-Spare-Festplatte ersetzt wurde, müssen Sie die Hot-Spare-Festplatte möglicherweise nach dem Ersetzen der ausgefallenen Festplatte abtrennen. Weitere Informationen zum Abtrennen von Hot-Spares finden Sie unter „Aktivieren und Deaktivieren von Hot-Spares im Speicher-Pool“ auf Seite 99.

Weitere Informationen zum Austauschen von Geräten finden Sie unter „Abhilfe bei Nichtverfügbarkeit eines Geräts“ auf Seite 309 sowie „Ersetzen oder Reparieren eines beschädigten Geräts“ auf Seite 311.

Zuweisen von Hot-Spares im Speicher-Pool

Mithilfe der Hot-Spare-Funktion können Sie Datenträger ermitteln, die zum Ersetzen eines ausgefallenen bzw. fehlerhaften Geräts in einem bzw. mehreren Speicher-Pools verwendet werden können. Das Vorsehen eines Geräts als *Hot-Spare* bedeutet, dass das Gerät im Pool nicht aktiv ist, sondern nur dazu dient, ein ausgefallenes Gerät im Pool automatisch zu ersetzen.

Datenspeichergeräte können mit den folgenden Methoden als Hot-Spares vorgesehen werden:

- bei der Erstellung eines Pools mit dem Befehl `zpool create`,
- nach der Erstellung eines Pools mit dem Befehl `zpool add`,
- Hot-Spares können von mehreren Pools gemeinsam genutzt werden, jedoch nicht von mehreren Pools verschiedener Systeme.

Das folgende Beispiel zeigt, wie Geräte bei der Erstellung des Pools als Hot-Spares zugewiesen werden:

```
# zpool create trinity mirror c1t1d0 c2t1d0 spare c1t2d0 c2t2d0
# zpool status trinity
  pool: trinity
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        trinity     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
        spares
          c1t2d0    AVAIL
          c2t2d0    AVAIL

errors: No known data errors
```

Das folgende Beispiel zeigt, wie Hot-Spares zugewiesen werden, indem sie zu einem Pool hinzugefügt werden, nachdem dieser erstellt wurde:

```
# zpool add neo spare c5t3d0 c6t3d0
# zpool status neo
  pool: neo
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        neo         ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c3t3d0  ONLINE       0     0     0
            c4t3d0  ONLINE       0     0     0
        spares
          c5t3d0    AVAIL
```

```
          c6t3d0    AVAIL

errors: No known data errors
```

Hot-Spares können mit dem Befehl `zpool remove` aus einem Speicher-Pool entfernt werden. Beispiel:

```
# zpool remove zeepool c2t3d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
        spares
          c1t3d0    AVAIL

errors: No known data errors
```

Ein Hot-Spare kann nur entfernt werden, wenn es gerade nicht vom Speicher-Pool verwendet wird.

Beachten Sie beim Arbeiten mit ZFS-Hot-Spares Folgendes:

- Zurzeit können mit dem Befehl `zpool remove` nur Hot-Spares, Cache-Geräte und Protokolliergeräte entfernt werden.
- Wenn eine Festplatte als Hot-Spare hinzugefügt werden soll, muss die Kapazität des Hot-Spares der Kapazität der größten Festplatte im Pool entsprechen oder größer sein. Sie können Festplatten mit geringerer Kapazität zwar hinzufügen, sollten jedoch beachten, dass bei Aktivierung der Festplatte mit der geringeren Kapazität (automatisch oder mithilfe des Befehls `zpool replace`) der Vorgang mit einer Fehlermeldung wie der Folgenden abbricht:

  `cannot replace` *disk3* `with` *disk4*`: device is too small`

Aktivieren und Deaktivieren von Hot-Spares im Speicher-Pool

Hot-Spares können mit einer der folgenden Methoden aktiviert werden:

- Manueller Austausch – Mithilfe des Befehls `zpool replace` können Sie ein ausgefallenes Gerät im Speicher-Pool durch ein Hot-Spare ersetzen.
- Automatischer Austausch – Wenn ein Fehler erkannt wird, überprüft ein FMA-Agent den Pool, um festzustellen, ob dieser verfügbare Hot-Spares enthält. Wenn dies der Fall ist, ersetzt er das fehlerhafte Gerät durch ein verfügbares Ersatzgerät.

  Falls ein gerade verwendetes Hot-Spare ausfällt, trennt der FMA-Agent das Ersatzgerät ab und macht den Austausch somit rückgängig. Dann versucht der Agent, das Datenspeichergerät mit einem anderen Hot-Spare zu ersetzen, falls dies möglich ist. Diese

Funktion ist gegenwärtig Beschränkungen unterworfen, da das ZFS-Diagnoseprogramm nur Fehler generiert, wenn ein Datenspeichergerät aus dem System verschwindet.

Wenn Sie ein ausgefallenes Datenspeichergerät physisch durch ein aktives Ersatzgerät ersetzen, können Sie das ursprüngliche Gerät wieder aktivieren, indem Sie das Ersatzgerät mithilfe des Befehls `zpool detach` abtrennen. Wenn Sie die Eigenschaft `autoreplace` des Pools auf on setzen, wird das Ersatzgerät automatisch getrennt und wieder in den Ersatzgeräte-Pool zurückgeführt, sobald das neue Datenspeichergerät eingesetzt und die Inbetriebnahme abgeschlossen ist.

Mit dem Befehl `zpool replace` können Sie ein Datenspeichergerät manuell durch ein Hot-Spare ersetzen. Siehe Beispiel 4–8.

Ein fehlerhaftes Datenspeichergerät wird automatisch ausgetauscht, wenn ein Hot-Spare verfügbar ist. Beispiel:

```
# zpool status -x
  pool: zeepool
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: resilver completed after 0h0m with 0 errors on Mon Jan 11 10:20:35 2010
config:

        NAME          STATE     READ WRITE CKSUM
        zeepool       DEGRADED     0     0     0
          mirror-0    DEGRADED     0     0     0
            c1t2d0    ONLINE       0     0     0
            spare-1   DEGRADED     0     0     0
              c2t1d0  UNAVAIL      0     0     0  cannot open
              c2t3d0  ONLINE       0     0     0  88.5K resilvered
        spares
          c2t3d0      INUSE     currently in use

errors: No known data errors
```

Derzeit können Sie ein Hot-Spare wie folgt deaktivieren:

- Durch Entfernen des Hot-Spares aus dem Speicher-Pool.
- Durch Trennen eines Hot-Spares, nachdem eine ausgefallene Festplatte physisch ersetzt wurde. Siehe Beispiel 4–9.
- Durch temporäres oder dauerhaftes ?Einlagern“ (Swap-in) des Hot-Spares. Siehe Beispiel 4–10.

BEISPIEL 4–8 Manuelles Ersetzen einer Festplatte durch ein Hot-Spare

In diesem Beispiel wird der Befehl `zpool replace` verwendet, um die Festplatte `c2t1d0` durch das Hot-Spare `c2t3d0` zu ersetzen.

BEISPIEL 4–8 Manuelles Ersetzen einer Festplatte durch ein Hot-Spare *(Fortsetzung)*

```
# zpool replace zeepool c2t1d0 c2t3d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Wed Jan 20 10:00:50 2010
config:

        NAME          STATE     READ WRITE CKSUM
        zeepool       ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c1t2d0    ONLINE       0     0     0
            spare-1   ONLINE       0     0     0
              c2t1d0  ONLINE       0     0     0
              c2t3d0  ONLINE       0     0     0  90K resilvered
        spares
          c2t3d0      INUSE     currently in use

errors: No known data errors
```

Dann trennen Sie die Festplatte `c2t1d0`.

```
# zpool detach zeepool c2t1d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Wed Jan 20 10:00:50 2010
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0  90K resilvered

errors: No known data errors
```

BEISPIEL 4–9 Trennen eines Hot-Spares, nachdem eine ausgefallene Festplatte physisch ersetzt wurde

In diesem Beispiel wird die ausgefallene Festplatte (`c2t1d0`) physisch ersetzt, und ZFS wird benachrichtig. Dazu wird der Befehl `zpool replace` verwendet.

```
# zpool replace zeepool c2t1d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Wed Jan 20 10:08:44 2010
config:

        NAME          STATE     READ WRITE CKSUM
        zeepool       ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c1t2d0    ONLINE       0     0     0
            spare-1   ONLINE       0     0     0
              c2t3d0  ONLINE       0     0     0  90K resilvered
```

BEISPIEL 4–9 Trennen eines Hot-Spares, nachdem eine ausgefallene Festplatte physisch ersetzt wurde *(Fortsetzung)*

```
            c2t1d0  ONLINE       0     0     0
        spares
          c2t3d0      INUSE     currently in use

errors: No known data errors
```

Danach können Sie den Befehl `zpool detach` verwenden, um das Hot-Spare in den Ersatzgeräte-Pool zurückzuführen. Beispiel:

```
# zpool detach zeepool c2t3d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed with 0 errors on Wed Jan 20 10:08:44 2010
config:

        NAME               STATE     READ WRITE CKSUM
        zeepool            ONLINE       0     0     0
          mirror           ONLINE       0     0     0
            c1t2d0         ONLINE       0     0     0
            c2t1d0         ONLINE       0     0     0
        spares
          c2t3d0           AVAIL

errors: No known data errors
```

BEISPIEL 4–10 Trennen einer ausgefallenen Festplatte und Verwenden des Hot-Spares

Wenn Sie eine ausgefallene Festplatte durch temporäres oder dauerhaftes Einlagern des Hot-Spares, das die ausgefallene Festplatte gerade ersetzt, ersetzen möchten, trennen Sie die ursprüngliche (ausgefallene) Festplatte. Wenn die ausgefallene Festplatte schließlich ersetzt ist, können Sie sie wieder in den Speicher-Pool zurückführen, wo sie als Ersatzfestplatte verfügbar ist. Beispiel:

```
# zpool status zeepool
  pool: zeepool
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using ’zpool online’.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: resilver in progress for 0h0m, 70.47% done, 0h0m to go
config:

        NAME          STATE     READ WRITE CKSUM
        zeepool       DEGRADED     0     0     0
          mirror-0    DEGRADED     0     0     0
            c1t2d0    ONLINE       0     0     0
            spare-1   DEGRADED     0     0     0
              c2t1d0  UNAVAIL      0     0     0  cannot open
              c2t3d0  ONLINE       0     0     0  70.5M resilvered
        spares
```

BEISPIEL 4–10 Trennen einer ausgefallenen Festplatte und Verwenden des Hot-Spares *(Fortsetzung)*

```
          c2t3d0      INUSE     currently in use

errors: No known data errors
# zpool detach zeepool c2t1d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Wed Jan 20 13:46:46 2010
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0  70.5M resilvered

errors: No known data errors
(Original failed disk c2t1d0 is physically replaced)
# zpool add zeepool spare c2t1d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Wed Jan 20 13:48:46 2010
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0  70.5M resilvered
        spares
          c2t1d0    AVAIL

errors: No known data errors
```

Eigenschaften von ZFS-Speicher-Pools

Mit dem Befehl `zpool get` können Sie Informationen zu den Pool-Eigenschaften abrufen. Beispiel:

```
# zpool get all mpool
NAME  PROPERTY       VALUE       SOURCE
pool  size           68G         -
pool  capacity       0%          -
pool  altroot        -           default
pool  health         ONLINE      -
pool  guid           601891032394735745  default
pool  version        22          default
pool  bootfs         -           default
pool  delegation     on          default
pool  autoreplace    off         default
pool  cachefile      -           default
```

```
pool  failmode       wait       default
pool  listsnapshots  on         default
pool  autoexpand     off        default
pool  free           68.0G      -
pool  allocated      76.5K      -
```

Mit dem Befehl `zpool set` lassen sich die Pool-Eigenschaften festlegen. Beispiel:

```
# zpool set autoreplace=on mpool
# zpool get autoreplace mpool
NAME  PROPERTY     VALUE    SOURCE
mpool autoreplace  on       default
```

TABELLE 4–1 Beschreibungen der Eigenschaften für ZFS-Pools

Eigenschaft	Typ	Standardwert	Beschreibung
`allocated`	Zeichenkette	entf.	Schreibgeschützter Wert, der die Menge des belegten Speicherplatzes im Pool angibt, der physisch zugewiesen ist.
`altroot`	Zeichenkette	`off`	Das alternative Root-Verzeichnis. Ist diese Eigenschaft gesetzt, wird dieses Verzeichnis jedem Einhängepunkt innerhalb des Pools vorangestellt. Diese Eigenschaft ist nützlich, um einen unbekannten Pool zu untersuchen, wenn die Einhängepunkte nicht vertrauenswürdig sind oder wenn in einer alternativen Boot-Umgebung die üblichen Pfade nicht gültig sind.
`autoreplace`	Boolesch	`off`	Regelt den automatischen Austausch von Speichergeräten. Wenn diese Eigenschaft auf `off` gesetzt ist, muss das Auswechseln von Speichergeräten mithilfe des Befehls zpool replace `eingeleitet werden.` Wenn diese Eigenschaft auf `on` gesetzt ist, wird das neue Speichergerät an der physischen Adresse des vorherigen Speichergeräts im Pool automatisch formatiert und in den Pool eingebunden. Die Abkürzung der Eigenschaft lautet `replace`.
`bootfs`	Boolesch	entf.	Das boot-fähige Standard-Dataset für den Root-Pool. Diese Eigenschaft wird in der Regel vom Installations- bzw. Upgrade-Programm gesetzt.
`cachefile`	Zeichenkette	entf.	Mit dieser Eigenschaft wird bestimmt, wo Pool-Konfigurationsinformationen im Cache gespeichert werden. Alle Pools im Cache werden beim Booten des Systems automatisch importiert. Installations- und Cluster-Umgebungen können jedoch erfordern, dass diese Informationen an anderer Stelle im Cache gespeichert werden, sodass Pools nicht automatisch importiert werden. Sie können diese Eigenschaft so einstellen, dass Poolkonfigurationen an einer anderen Stelle im Cache-Speicher abgelegt werden. Diese Informationen können später mithilfe des Befehls `zpool import - c` importiert werden. Bei den meisten ZFS-Konfigurationen wird diese Eigenschaft nicht verwendet.

TABELLE 4–1 Beschreibungen der Eigenschaften für ZFS-Pools *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`capacity`	Zahl	entf.	Schreibgeschützter Wert, der die Menge des belegten Speicherplatzes im Pool als Verhältnis zur Gesamtkapazität in Prozent angibt. Die Abkürzung der Eigenschaft lautet `cap`.
`delegation`	Boolesch	`on`	Bestimmt, ob einem Benutzer ohne ausreichende Berechtigungen die für das Dataset festgelegten Zugriffsrechte erteilt werden können. Weitere Informationen finden Sie in Kapitel 9, „Delegierte ZFS-Administration“.
`failmode`	Zeichenkette	`wait`	Steuert das Systemverhalten, wenn ein schwerwiegender Pool-Ausfall auftritt. Dieser Umstand ist in der Regel das Ergebnis des Konnektivitätsverlusts eines oder mehrerer zugrunde liegender Speichergeräte oder eines Ausfalls sämtlicher Geräte im Pool. In solch einem Fall wird das Verhalten durch einen der folgenden Werte bestimmt: ■ `wait` – Blockiert sämtliche E/A-Anforderungen, bis die Gerätekonnektivität wiederhergestellt ist und die Fehler mit dem Befehl `zpool clear` zurückgesetzt wurden. In diesem Zustand werden die E/A-Vorgänge, die den Pool betreffen, blockiert. Lesevorgänge können jedoch eventuell ausgeführt werden. Ein Pool bleibt im Wartezustand, bis das Geräteproblem behoben ist. ■ `continue` – Gibt bei jeder neuen E/A-Schreibanforderung einen E/A-Fehler zurück, lässt aber Lesezugriffe auf die übrigen fehlerfreien Speichergeräte zu. Schreibanforderungen, die noch an die Festplatte übermittelt werden müssen, werden blockiert. Nach Wiederherstellung der Verbindung oder Ersetzen des Geräts müssen die Fehler mit dem Befehl `zpool clear` zurückgesetzt werden. ■ `panic` – Gibt eine Meldung an die Konsole aus und generiert einen Systemabsturz-Speicherabzug.
`free`	Zeichenkette	entf.	Schreibgeschützter Wert, der die Menge von Datenblöcken im Pool angibt, die nicht zugewiesen sind.
`guid`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft und eindeutige Kennzeichnung des Pools.
`health`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft, die den aktuellen Zustand des Pools angibt. Dabei bestehen folgende Möglichkeiten: ONLINE, DEGRADED, FAULTED, OFFLINE, REMOVED oder UNAVAIL

TABELLE 4–1 Beschreibungen der Eigenschaften für ZFS-Pools *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`listsnapshots`	Zeichenkette	`on`	Kontrolliert, ob Snapshot-Informationen, die mit diesem Pool in Verbindung stehen, mit dem Befehl `zfs list` angezeigt werden. Wenn diese Eigenschaft deaktiviert wird, können die Snapshot-Informationen mit dem Befehl `zfs list -t snapshot` angezeigt werden.
`size`	Zahl	entf.	Schreibgeschützte Eigenschaft, die die Gesamtkapazität des Speicher-Pools angibt.
`version`	Zahl	entf.	Die aktuelle Version des Pools. Zum Aktualisieren von Pools wird grundsätzlich die Methode mit dem Befehl `zpool upgrade` empfohlen. Diese Eigenschaft bietet sich dann an, wenn aus Gründen der Abwärtskompatibilität eine spezifische Version benötigt wird. Diese Eigenschaft kann auf jede Zahl zwischen 1 und der aktuellen Version laut Ausgabe des Befehls `zpool upgrade -v` gesetzt werden.

Abfragen des Status von ZFS-Speicher-Pools

Mithilfe des Befehls `zpool list` können Sie mit unterschiedlichen Methoden Informationen zum Pool-Status abrufen. Die verfügbaren Informationen unterteilen sich im Allgemeinen in drei Kategorien: grundlegende Informationen zur Auslastung, E/A-Statistiken und Informationen zum Funktionsstatus. In diesem Abschnitt werden alle drei Kategorien dieser Informationen zu Speicher-Pools behandelt.

- „Anzeigen von Informationen zu ZFS-Speicher-Pools“ auf Seite 106
- „Anzeigen von E/A-Statistiken für ZFS-Speicher-Pools “ auf Seite 110
- „Ermitteln des Funktionsstatus von ZFS-Speicher-Pools“ auf Seite 112

Anzeigen von Informationen zu ZFS-Speicher-Pools

Mit dem Befehl `zpool list` können Sie grundlegende Pool-Informationen anzeigen.

Anzeigen von Informationen zu allen Speicher-Pools oder einem bestimmten Pool

Ohne Argumente werden mithilfe des Befehls `zpool list` folgende Informationen für alle Pools des Systems angezeigt:

```
# zpool list
NAME                    SIZE    ALLOC   FREE    CAP  HEALTH     ALTROOT
tank                   80.0G   22.3G   47.7G    28%  ONLINE     -
dozer                   1.2T    384G    816G    32%  ONLINE     -
```

Diese Befehlsausgabe zeigt folgende Informationen an:

`NAME` Der Name des Pools.

`SIZE` Die Gesamtkapazität des Pools entspricht der Summe der Speicherkapazität aller virtuellen Geräte der obersten Hierarchieebene.

`ALLOC` Der von allen Datasets und internen Metadaten belegte physische Speicherplatz. Bitte beachten Sie, dass sich dieser Wert von der auf Dateisystemebene gemeldeten Festplattenkapazität unterscheidet.

Weitere Informationen zum Ermitteln des verfügbaren Dateisystemspeicherplatzes finden Sie unter „Berechnung von ZFS-Festplattenkapazität“ auf Seite 62.

`FREE` Der Wert des nicht belegten Speicherplatzes im Pool.

`CAP (CAPACITY)` Der Wert der belegten Festplattenkapazität als Verhältnis zur Gesamtkapazität, in Prozent.

`HEALTH` Der gegenwärtige Funktionsstatus des Pools.

Weitere Informationen zum Pool-Status finden Sie unter „Ermitteln des Funktionsstatus von ZFS-Speicher-Pools“ auf Seite 112.

`ALTROOT` Das alternative Root-Verzeichnis des Pools (falls vorhanden).

Weitere Informationen zu Speicher-Pools mit alternativem Root-Verzeichnis finden Sie unter „Verwenden von ZFS-Speicher-Pools mit alternativem Root-Verzeichnis“ auf Seite 296.

Durch Angabe eines Pool-Namens können Sie sich auch Statistiken zu diesem bestimmten Pool anzeigen lassen. Beispiel:

```
# zpool list tank
NAME                    SIZE    ALLOC   FREE    CAP  HEALTH     ALTROOT
tank                   80.0G    22.3G   47.7G   28%  ONLINE     -
```

Anzeigen spezifischer Speicher-Pool-Statistikinformationen

Mithilfe der Option `-o` können Sie sich spezifische Statistikinformationen anzeigen lassen. Auch können Sie mithilfe dieser Option benutzerdefinierte Berichte erstellen oder sich gewünschte Informationen anzeigen lassen. Mit der folgenden Syntax wird beispielsweise nur der Name und die Speicherkapazität jedes Pools angezeigt:

```
# zpool list -o name,size
NAME                    SIZE
tank                   80.0G
dozer                   1.2T
```

Die Spaltentitel sind unter „Anzeigen von Informationen zu allen Speicher-Pools oder einem bestimmten Pool“ auf Seite 106 erläutert.

Verwenden von Ausgaben von ZFS-Speicher-Pools für Skripten

Die Standardausgabe des Befehls `zpool list` dient der Lesbarkeit am Bildschirm und ist für Shell-Skripten nicht zu verwenden. Zur Unterstützung programmatischer Verwendungen dieses Befehls kann mithilfe der Option `-H` die Ausgabe der Spaltentitel unterdrückt werden, und die einzelnen Felder werden durch Leerzeichen statt durch Tabulatoren getrennt. So rufen Sie beispielsweise eine Liste aller Pool-Namen im System mithilfe der folgenden Syntax ab:

```
# zpool list -Ho name
tank
dozer
```

Hier ist ein weiteres Beispiel:

```
# zpool list -H -o name,size
tank   80.0G
dozer  1.2T
```

Anzeige des Befehlsprotokolls von ZFS-Speicher-Pools

ZFS protokolliert automatisch `zfs`- und `zpool`-Befehle, durch die Pool-Zustandsinformationen geändert werden. Diese Informationen können mit dem Befehl `zpool history` angezeigt werden.

Die folgende Syntax zeigt beispielsweise die Befehlsausgabe für den Root-Pool:

```
# zpool history
History for 'rpool':
2010-05-11.10:18:54 zpool create -f -o failmode=continue -R /a -m legacy -o
cachefile=/tmp/root/etc/zfs/zpool.cache rpool mirror c1t0d0s0 c1t1d0s0
2010-05-11.10:18:55 zfs set canmount=noauto rpool
2010-05-11.10:18:55 zfs set mountpoint=/rpool rpool
2010-05-11.10:18:56 zfs create -o mountpoint=legacy rpool/ROOT
2010-05-11.10:18:57 zfs create -b 8192 -V 2048m rpool/swap
2010-05-11.10:18:58 zfs create -b 131072 -V 1536m rpool/dump
2010-05-11.10:19:01 zfs create -o canmount=noauto rpool/ROOT/zfsBE
2010-05-11.10:19:02 zpool set bootfs=rpool/ROOT/zfsBE rpool
2010-05-11.10:19:02 zfs set mountpoint=/ rpool/ROOT/zfsBE
2010-05-11.10:19:03 zfs set canmount=on rpool
2010-05-11.10:19:04 zfs create -o mountpoint=/export rpool/export
2010-05-11.10:19:05 zfs create rpool/export/home
2010-05-11.11:11:10 zpool set bootfs=rpool rpool
2010-05-11.11:11:10 zpool set bootfs=rpool/ROOT/zfsBE rpool
```

Sie können auf Ihrem System eine ähnliche Ausgabe verwenden, um die ZFS-Befehle zu identifizieren, die bei der Behebung eines Fehlers ausgeführt wurden.

Das Verlaufsprotokoll weist folgende Merkmale auf:

- Das Protokoll kann nicht deaktiviert werden.
- Das Protokoll wird persistent gespeichert, es wird also neustartübergreifend geführt.
- Das Protokoll wird in Form eines Ringpuffers implementiert. Die Mindestgröße beträgt 128 KB. Die Maximalgröße beträgt 32 MB.
- Für kleinere Pools ist die maximale Größe auf 1 Prozent der Pool-Größe beschränkt, wobei die *Größe* zum Zeitpunkt der Pool-Erstellung bestimmt wird.
- Das Protokoll erfordert keine Verwaltung, d. h., eine Anpassung der Protokollgröße bzw. eine Änderung des Speicherorts sind nicht nötig.

Verwenden Sie zur Identifizierung des Befehlsprotokolls eines bestimten Speicher-Pools in etwa folgende Syntax:

```
# zpool history tank
History for 'tank':
2010-05-13.14:13:15 zpool create tank mirror c1t2d0 c1t3d0
2010-05-13.14:21:19 zfs create tank/snaps
2010-05-14.08:10:29 zfs create tank/ws01
2010-05-14.08:10:54 zfs snapshot tank/ws01@now
2010-05-14.08:11:05 zfs clone tank/ws01@now tank/ws01bugfix
```

Verwenden Sie die Option `-l` zum Anzeigen eines langen Formats mit Benutzernamen, Hostnamen und Angabe der Zone, in der der Vorgang ausgeführt wurde. Beispiel:

```
# zpool history -l tank
History for 'tank':
2010-05-13.14:13:15 zpool create tank mirror c1t2d0 c1t3d0 [user root on neo]
2010-05-13.14:21:19 zfs create tank/snaps [user root on neo]
2010-05-14.08:10:29 zfs create tank/ws01 [user root on neo]
2010-05-14.08:10:54 zfs snapshot tank/ws01@now [user root on neo]
2010-05-14.08:11:05 zfs clone tank/ws01@now tank/ws01bugfix [user root on neo]
```

Verwenden Sie die Option `-i` zum Anzeigen interner Ereignisinformationen, die bei der Diagnose behilflich sein können. Beispiel:

```
# zpool history -i tank
2010-05-13.14:13:15 zpool create -f tank mirror c1t2d0 c1t23d0
2010-05-13.14:13:45 [internal pool create txg:6] pool spa 19; zfs spa 19; zpl 4;...
2010-05-13.14:21:19 zfs create tank/snaps
2010-05-13.14:22:02 [internal replay_inc_sync txg:20451] dataset = 41
2010-05-13.14:25:25 [internal snapshot txg:20480] dataset = 52
2010-05-13.14:25:25 [internal destroy_begin_sync txg:20481] dataset = 41
2010-05-13.14:25:26 [internal destroy txg:20488] dataset = 41
2010-05-13.14:25:26 [internal reservation set txg:20488] 0 dataset = 0
2010-05-14.08:10:29 zfs create tank/ws01
2010-05-14.08:10:54 [internal snapshot txg:53992] dataset = 42
2010-05-14.08:10:54 zfs snapshot tank/ws01@now
2010-05-14.08:11:04 [internal create txg:53994] dataset = 58
2010-05-14.08:11:05 zfs clone tank/ws01@now tank/ws01bugfix
```

Anzeigen von E/A-Statistiken für ZFS-Speicher-Pools

Mit dem Befehl `zpool iostat` können Sie E/A-Statistikinformationen für einen Pool bzw. ein virtuelles Datenspeichergerät abrufen. Dieser Befehl zeigt ähnlich wie der Befehl `iostat` eine statische ?Momentaufnahme" aller E/A-Aktivitäten sowie aktualisierte Statistikinformationen für jedes definierte Zeitintervall an. Es werden die folgenden Statistikinformationen ausgegeben:

`alloc capacity` Die Kapazität der gegenwärtig im Pool bzw. Gerät gespeicherten Daten. Aufgrund interner Implementierungsaspekte unterscheidet sich dieser Wert geringfügig von der für die betreffenden Dateisysteme verfügbaren Festplattenkapazität.

Weitere Informationen zu Unterschieden zwischen Pool- und Dataset-Speicherplatz finden Sie in „Berechnung von ZFS-Festplattenkapazität" auf Seite 62.

`free capacity` Die im Pool bzw. Gerät verfügbare Festplattenkapazität. Wie in der `used`-Statistik unterscheidet sich dieser Wert geringfügig von der für Datasets verfügbaren Festplattenkapazität.

`read operations` Die Anzahl der zum Pool bzw. Gerät gesendeten E/A-Vorgänge einschließlich Metadaten-Anforderungen.

`write operations` Die Anzahl der zum Pool bzw. Gerät gesendeten E/A-Schreiboperationen.

`read bandwidth` Die Bandbreite aller Leseoperationen (einschließlich Metadaten) in Einheiten pro Sekunde.

`write bandwidth` Die Bandbreite aller Schreiboperationen in Einheiten pro Sekunde.

Anzeigen globaler Pool-E/A-Statistikinformationen

Ohne Optionen zeigt der Befehl `zpool iostat` die bisher aufgelaufenen Statistikinformationen für alle Pools im System seit dem Hochfahren des Systems an. Beispiel:

```
# zpool iostat
               capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
rpool       6.05G  61.9G      0      0    786    107
tank        31.3G  36.7G      4      1   296K  86.1K
----------  -----  -----  -----  -----  -----  -----
```

Da die angezeigten Statistikinformationen seit dem Hochfahren des Systems aufgelaufen sind, kann es sein, dass die Bandbreite bei relativ geringer Auslastung des Pools gering erscheint. Durch Angabe eines Zeitintervalls erhalten Sie ein realistischeres Bild der aktuellen Bandbreite. Beispiel:

```
# zpool iostat tank 2
               capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
tank        18.5G  49.5G      0    187      0  23.3M
tank        18.5G  49.5G      0    464      0  57.7M
tank        18.5G  49.5G      0    457      0  56.6M
tank        18.8G  49.2G      0    435      0  51.3M
```

In diesem Beispiel zeigt der Befehl alle zwei Sekunden lang die Statistikinformationen zur Auslastung für den Pool `tank` an, und zwar lange, bis Sie die Tastenkombination CTRL-C drücken. Als Alternative können Sie ein zusätzliches Argument (`count`) angeben, das die Ausführung des Befehls nach einer bestimmten Anzahl von Wiederholungen beendet. So gibt `zpool iostat 2 3` beispielsweise dreimal alle zwei Sekunden (also insgesamt sechs Sekunden lang) eine Übersicht aus. Wenn nur ein einziger Pool vorhanden ist, werden die Statistikinformationen in aufeinander folgenden Zeilen angezeigt. Sind mehrere Pools vorhanden, werden die Werte für die einzelnen Pools zur besseren Lesbarkeit durch gestrichelte Linien getrennt.

Anzeigen von E/A-Statistikinformationen zu virtuellen Geräten

Neben den globalen E/A-Statistikinformationen für einen Pool kann der Befehl `zpool iostat` außerdem E/A-Statistikinformationen für bestimmte virtuelle Datenspeichergeräte anzeigen. Mit diesem Befehl können Sie unverhältnismäßig langsame Datenspeichergeräte identifizieren oder die Verteilung der von ZFS generierten E/A-Vorgänge überprüfen. Zum Abrufen der vollständigen virtuellen Gerätestruktur sowie aller E/A-Statistikinformationen können Sie den Befehl `zpool iostat -v` nutzen. Beispiel:

```
# zpool iostat -v
               capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
rpool       6.05G  61.9G      0      0    785    107
  mirror    6.05G  61.9G      0      0    785    107
    c1t0d0s0    -      -      0      0    578    109
    c1t1d0s0    -      -      0      0    595    109
----------  -----  -----  -----  -----  -----  -----
tank        36.5G  31.5G      4      1   295K   146K
  mirror    36.5G  31.5G    126     45  8.13M  4.01M
    c1t2d0      -      -      0      3   100K   386K
    c1t3d0      -      -      0      3   104K   386K
----------  -----  -----  -----  -----  -----  -----
```

Bitte beachten Sie zwei wichtige Aspekte, wenn Sie E/A-Statistikinformationen für virtuelle Geräte abrufen:

- Erstens sind Statistikinformationen zur Belegung von Festplattenkapazität nur für virtuelle Geräte der obersten Hierarchieebene verfügbar. Die Art und Weise der Zuweisung von Festplattenkapazität bei virtuellen Geräten mit Datenspiegelung und RAID-Z ist implementierungsspezifisch und kann nicht in Form einer einzelnen Zahl ausgedrückt werden.
- Zweitens kann es sein, dass die einzelnen Werte nicht die erwartete Summe ergeben. Insbesondere sind Werte bei Geräten mit Datenspiegelung und RAID-Z nicht genau gleich. Diese Unterschiede sind besonders nach der Erstellung eines Pools bemerkbar, da ein großer Teil der E/A-Vorgänge infolge der Pool-Erstellung direkt auf den Datenträgern ausgeführt wird, was auf der Datenspiegelungsebene nicht berücksichtigt wird. Im Laufe der Zeit gleichen sich diese Werte allmählich an. Allerdings können auch defekte, nicht reagierende bzw. außer Betrieb genommen Geräte diese Symmetrie beeinträchtigen.

Bei der Untersuchung von Statistikinformationen zu virtuellen Geräten können Sie die gleichen Optionen (Zeitintervall und Zählparameter) verwenden.

Ermitteln des Funktionsstatus von ZFS-Speicher-Pools

ZFS bietet eine integrierte Methode zur Untersuchung der ordnungsgemäßen Funktion von Pools und Datenspeichergeräten. Der Funktionsstatus eines Pools wird aus dem Funktionsstatus aller seiner Datenspeichergeräte ermittelt. Diese Statusinformationen werden mit dem Befehl `zpool status` angezeigt. Außerdem werden potenzielle Pool- und Geräteausfälle von `fmd` gemeldet, an der Systemkonsole angezeigt und in der Datei `/var/adm/messages` protokolliert.

In diesem Abschnitt wird die Ermittlung des Funktionsstatus von Pools und Datenspeichergeräten erläutert. Dieses Kapitel enthält jedoch keine Informationen zur Reparatur fehlerhafter Pools bzw. Wiederherstellen des Normalbetriebs eines Pools. Weitere Informationen zur Fehlerbehebung und Datenwiederherstellung finden Sie in Kapitel 11, „Problembehebung und Pool-Wiederherstellung in Oracle Solaris ZFS“.

Datenspeichergeräte können sich in einem der folgenden Zustände befinden:

`ONLINE` — Das Gerät bzw. virtuelle Gerät arbeitet normal. Obwohl zeitweilige Übergangsfehler auftreten können, arbeitet das Gerät sonst einwandfrei.

`DEGRADED` — Am virtuellen Gerät ist ein Fehler aufgetreten, es funktioniert jedoch noch. Dieser Zustand tritt am Häufigsten auf, wenn in einer RAID-Z-Konfiguration ein oder mehrere Datenspeichergeräte nicht mehr verfügbar sind. Die Fehlertoleranz des Pools kann beeinträchtigt werden, da ein Ausfall eines weiteren Geräts nicht behebbar sein könnte.

`FAULTED` — Auf das Gerät bzw. virtuelle Gerät kann nicht zugegriffen werden. Dieser Status zeigt normalerweise einen Totalausfall des Geräts an, bei dem ZFS mit dem Gerät

keine Daten mehr austauschen kann. Wenn sich ein virtuelles Gerät der obersten Hierarchieebene in diesem Status befindet, kann auf den gesamten Pool nicht mehr zugegriffen werden.

`OFFLINE` Das Gerät wurde vom Administrator außer Betrieb genommen.

`UNAVAIL` Mit dem Gerät bzw. virtuellen Gerät kann nicht kommuniziert werden. In manchen Fällen gehen Pools mit Geräten im Status `UNAVAIL` in den Status `DEGRADED`. Wenn sich ein virtuelles Gerät der obersten Hierarchieebene im Status `UNAVAIL` befindet, können von diesem Pool keine Daten abgerufen werden.

`REMOVED` Das Gerät wurde bei laufendem Systembetrieb physisch ausgebaut. Die Erkennung ausgebauter Geräte ist von der jeweiligen Hardware abhängig und wird möglicherweise nicht auf allen Plattformen unterstützt.

Der Funktionsstatus eines Pools wird aus dem Funktionsstatus aller seiner Datenspeichergeräte der obersten Hierarchieebene ermittelt. Befinden sich alle virtuellen Geräte eines Pools im Status `ONLINE`, besitzt der Pool ebenfalls den Status `ONLINE`. Befindet sich eines der virtuellen Geräte eines Pools im Status `DEGRADED` bzw. `UNAVAIL`, besitzt der Pool den Status `DEGRADED`. Wenn sich ein virtuelles Gerät der obersten Hierarchieebene im Status `FAULTED` bzw. `OFFLINE` befindet, besitzt der Pool den Status `FAULTED`. Auf einen Pool im Status `FAULTED` kann nicht zugegriffen werden. Es können erst wieder Daten abgerufen werden, wenn erforderliche Datenspeichergeräte verbunden bzw. repariert werden. Ein Pool im Status `DEGRADED` bleibt zwar weiterhin aktiv, es kann jedoch sein, dass nicht das gleiche Datenredundanz- bzw. Datendurchsatzniveau wie bei einem ordnungsgemäßen Funktionieren des Pools erreicht wird.

Grundlegender Funktionsstatus eines Speicher-Pools

Sie können den Funktionsstatus eines Pools mithilfe des Befehls `zpool status` wie folgt rasch abrufen:

```
# zpool status -x
all pools are healthy
```

Bestimmte Pools können geprüft werden, indem der Pool-Name in der Befehlssyntax angegeben wird. Pools, die sich nicht im Status `ONLINE` befinden, sollten auf potenzielle Probleme untersucht werden (siehe folgender Abschnitt).

Ausführliche Informationen zum Funktionsstatus

Mithilfe der Option `-v` können Sie ausführlichere Informationen zum Funktionsstatus abrufen. Beispiel:

```
# zpool status -v tank
  pool: tank
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
```

```
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: scrub completed after 0h0m with 0 errors on Wed Jan 20 15:13:59 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  UNAVAIL      0     0     0  cannot open

errors: No known data errors
```

Diese Ausgabe enthält eine vollständige Beschreibung darüber, warum sich der Pool in seinem gegenwärtigen Funktionsstatus befindet. Es findet sich auch eine lesbare Erläuterung des Problems und ein Verweis auf einen Artikel in der Sun Knowledge Base, wenn Sie weitere Informationen dazu benötigen. Die Artikel der Sun Knowledge Base enthalten die aktuellsten Informationen zur Behebung eines bestimmten Problems. Mithilfe der aufgeführten ausführlichen Konfigurationsinformationen sollten Sie feststellen können, welches Datenspeichergerät defekt ist und wie Sie den Pool reparieren können.

Im vorherigen Beispiel muss das defekte Datenspeichergerät ausgetauscht werden. Nach dem Austauschen des Geräts können Sie es mit dem Befehl `zpool online` wieder in Betrieb nehmen. Beispiel:

```
# zpool online tank c1t0d0
Bringing device c1t0d0 online
# zpool status -x
all pools are healthy
```

Wenn die Eigenschaft `autoreplace` aktiviert ist, müssen Sie das ausgetauschte Gerät möglicherweise nicht in Betrieb nehmen.

Wenn in einem Pool ein außer Betrieb genommenes Gerät vorhanden ist, kann es mithilfe der vom Befehl ausgegebenen Informationen identifiziert werden. Beispiel:

```
# zpool status -x
  pool: tank
 state: DEGRADED
status: One or more devices has been taken offline by the administrator.
        Sufficient replicas exist for the pool to continue functioning in a
        degraded state.
action: Online the device using 'zpool online' or replace the device with
        'zpool replace'.
 scrub: resilver completed after 0h0m with 0 errors on Wed Jan 20 15:15:09 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  OFFLINE      0     0     0  48K resilvered
```

```
errors: No known data errors
```

In den Spalten READ und WRITE wird die Anzahl der für das betreffende Gerät gezählten E/A-Fehler angezeigt, und die Spalte CKSUM enthält die Anzahl der am Gerät aufgetretenen nicht behebbaren Prüfsummenfehler. Da beide Fehlerzähler auf einen wahrscheinlichen Geräteausfall hinweisen, sollten Sie Maßnahmen zur Behebung dieses Problems einleiten. Wenn für ein virtuelles Gerät der obersten Hierarchieebene Fehlerwerte angezeigt werden, die nicht gleich null sind, kann es sein, dass auf Daten teilweise nicht mehr zugegriffen werden kann

Das Feld `errors:` weist auf bekannte Datenfehler hin.

In der Befehlsausgabe des vorherigen Beispiels verursacht das außer Betrieb genommene Gerät keine Datenfehler.

Weitere Informationen zum Auffinden von Fehlern und Reparieren fehlerhafter Pools und Daten finden Sie in Kapitel 11, „Problembehebung und Pool-Wiederherstellung in Oracle Solaris ZFS“.

Migrieren von ZFS-Speicher-Pools

Gelegentlich kann es vorkommen, dass Speicher-Pools zwischen Systemen transferiert werden müssen. Dafür müssen Sie die Datenspeichergeräte aus dem ursprünglichen System herausnehmen und an das neue System anschließen. Dies kann durch Neuverkabelung der betreffenden Geräte bzw. die Verwendung von Geräten mit mehreren Anschlüssen, z. B. Geräte in einem Speichernetzwerk (SAN), bewerkstelligt werden. Mit ZFS können Sie einen Pool aus einem System exportieren und in das Zielsystem importieren. Dies ist auch möglich, wenn beide Rechnerarchitekturen unterschiedliche Bitbreiten besitzen. Informationen zum Replizieren bzw. Migrieren von Dateisystemen zwischen verschiedenen Speicher-Pools, die auf unterschiedlichen Rechnersystemen installiert sind, finden Sie unter „Senden und Empfangen von ZFS-Daten“ auf Seite 234.

Vorbereiten der Migration eines ZFS-Speicher-Pools

Speicher-Pools sollten explizit exportiert werden, um anzuzeigen, dass sie zur Migration bereit sind. Bei diesem Vorgang werden ungeschriebene Daten auf die Festplatte ausgespeichert und Daten auf die Festplatte geschrieben, wodurch angezeigt wird, dass der Export abgeschlossen ist. Anschließend werden alle Informationen, die den Pool betreffen, aus dem System entfernt.

Wenn Sie den Pool nicht explizit exportieren, sondern die Datenträger manuell entfernen, kann der resultierende Pool trotzdem noch in ein anderes System importiert werden. Es kann jedoch sein, dass die in den allerletzten Sekunden ausgeführten Datentransaktionen verloren gehen und der betreffende Pool auf dem ursprünglichen System als fehlerhaft angezeigt wird, da die Datenspeichergeräte nicht mehr vorhanden sind. Standardmäßig kann ein Pool, der nicht explizit exportiert wurde, nicht vom Zielsystem importiert werden. Dies ist erforderlich, um zu verhindern, dass versehentlich ein aktiver Pool importiert wird, der Speicherplatz enthält, der über das Netzwerk zugänglich ist und noch von einem anderem System belegt wird.

Exportieren eines ZFS-Speicher-Pools

Speicher-Pools können mit dem Befehl `zpool export` exportiert werden. Beispiel:

```
# zpool export tank
```

Vor dem Fortfahren versucht der Befehl alle innerhalb des Pools eingehängten Dateisysteme auszuhängen. Falls das Aushängen von Dateisystemen fehlschlägt, können Sie mithilfe der Option `-f` ein Aushängen erzwingen. Beispiel:

```
# zpool export tank
cannot unmount '/export/home/eschrock': Device busy
# zpool export -f tank
```

Nach der Ausführung dieses Befehls ist der Pool `tank` im System nicht mehr sichtbar.

Falls Datenspeichergeräte zum Zeitpunkt des Exports nicht verfügbar sind, können die betreffenden Geräte nicht als ?sauber“ exportiert eingestuft werden. Wenn ein solches Datenspeichergerät später ohne die funktionierenden Datenspeichergeräte mit einem System verbunden wird, erscheint das betreffende Gerät als potenziell aktiv.”

Wenn im Pool ZFS-Volumes vorhanden sind, kann der Pool auch nicht mithilfe der Option `-f` exportiert werden. Wenn Sie einen Pool mit einem ZFS-Volume exportieren möchten, müssen Sie zunächst sicherstellen, dass das Volume nicht von aktiven Ressourcen belegt wird.

Weitere Informationen zu ZFS-Volumes finden Sie unter „ZFS-Volumes“ auf Seite 287.

Ermitteln verfügbarer Speicher-Pools für den Import

Nachdem ein Pool (durch expliziten Export oder erzwungenes Entfernen von Datenspeichergeräten) aus einem System entfernt wurde, müssen die betreffenden Geräte mit dem Zielsystem verbunden werden. ZFS kann Situationen bewältigen, in denen nur einige der Geräte verfügbar sind. Der Erfolg der Pool-Migration hängt jedoch von der Funktionstüchtigkeit der Geräte ab. Die Geräte müssen jedoch nicht notwendigerweise unter dem gleichen Gerätenamen verbunden werden. ZFS erkennt verschobene bzw. umbenannte Geräte und passt die Konfiguration entsprechend an. Führen Sie zum Ermitteln importierbarer Pools den Befehl `zpool export` aus. Beispiel:

```
# zpool import
 pool: tank
    id: 11809215114195894163
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        tank        ONLINE
          mirror-0  ONLINE
            c1t0d0  ONLINE
            c1t1d0  ONLINE
```

In diesem Beispiel kann der Pool `tank` in ein Zielsystem importiert werden. Jeder Pool wird durch einen Namen und einen eindeutigen nummerischen Bezeichner identifiziert. Wenn mehrere Pools mit dem gleichen Namen für den Import verfügbar sind, können sie mithilfe des nummerischen Bezeichners unterschieden werden.

Ebenso wie die Ausgabe des Befehls `zpool status` verweist die Ausgabe des Befehls `zpool import` auf einen Artikel der Sun Knowledge Base, der die aktuellsten Informationen und Reparaturhinweise zu Problemen enthält, die das Importieren von Pools verhindern. In diesem Fall kann das Importieren eines Pools erzwungen werden. Das Importieren eines Pools, der gegenwärtig über Netzwerkzugriff von einem anderen System verwendet wird, kann Daten beschädigen und auf beiden Systemen zu Abstürzen führen, wenn diese Systeme Daten auf das gleiche Datenspeichergerät schreiben. Wenn einige Geräte eines Pools nicht verfügbar sind, zum Bereitstellen eines funktionierenden Pools jedoch genügend Redundanzdaten vorhanden sind, geht der Pool in den Status `DEGRADED`. Beispiel:

```
# zpool import
  pool: tank
    id: 11809215114195894163
 state: DEGRADED
status: One or more devices are missing from the system.
action: The pool can be imported despite missing or damaged devices.  The
        fault tolerance of the pool may be compromised if imported.
   see: http://www.sun.com/msg/ZFS-8000-2Q
config:

        NAME        STATE     READ WRITE CKSUM
```

```
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t0d0  UNAVAIL      0     0     0  cannot open
            c1t3d0  ONLINE       0     0     0
```

In diesem Beispiel ist der erste Datenträger beschädigt oder nicht vorhanden, der Pool kann aber trotzdem importiert werden, da die gespiegelten Daten noch verfügbar sind. Wenn zuviele fehlerhafte Datenspeichergeräte vorhanden sind bzw. Geräte fehlen, kann der Pool nicht importiert werden. Beispiel:

```
# zpool import
  pool: dozer
    id: 9784486589352144634
 state: FAULTED
action: The pool cannot be imported. Attach the missing
        devices and try again.
   see: http://www.sun.com/msg/ZFS-8000-6X
config:
        raidz1-0       FAULTED
          c1t0d0       ONLINE
          c1t1d0       FAULTED
          c1t2d0       ONLINE
          c1t3d0       FAULTED
```

In diesem Beispiel fehlen in einem virtuellen RAID-Z-Gerät zwei Datenträger, was bedeutet, dass zum Rekonstruieren des Pools nicht genügend Redundanz verfügbar ist. In einigen Fällen kann es auch sein, dass zum Ermitteln der vollständigen Konfiguration nicht genügend Datenspeichergeräte vorhanden sind. In einem solchen Fall kann ZFS nicht bestimmen, welche anderen Geräte zu diesem Pool gehört haben, obwohl ZFS diesbezüglich so viele Informationen wie möglich meldet. Beispiel:

```
# zpool import
pool: dozer
    id: 9784486589352144634
 state: FAULTED
status: One or more devices are missing from the system.
action: The pool cannot be imported. Attach the missing
        devices and try again.
   see: http://www.sun.com/msg/ZFS-8000-6X
config:
        dozer          FAULTED   missing device
          raidz1-0     ONLINE
            c1t0d0     ONLINE
            c1t1d0     ONLINE
            c1t2d0     ONLINE
            c1t3d0     ONLINE
Additional devices are known to be part of this pool, though their
exact configuration cannot be determined.
```

Importieren von ZFS-Speicher-Pools aus anderen Verzeichnissen

Standardmäßig durchsucht der Befehl `zpool import` nur im Verzeichnis `/dev/dsk` enthaltene Datenspeichergeräte. Wenn Geräte in einem anderen Verzeichnis vorhanden sind oder Sie Pools verwenden, die durch Dateien gesichert sind, müssen Sie mithilfe der Option `-d` andere Verzeichnisse durchsuchen. Beispiel:

```
# zpool create dozer mirror /file/a /file/b
# zpool export dozer
# zpool import -d /file
  pool: dozer
    id: 7318163511366751416
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer        ONLINE
          mirror-0   ONLINE
            /file/a  ONLINE
            /file/b  ONLINE
# zpool import -d /file dozer
```

Sie können die Option `-d` mehrmals angeben, wenn Datenspeichergeräte in mehreren Verzeichnissen vorhanden sind.

Importieren von ZFS-Speicher-Pools

Wenn ein Pool für den Import ermittelt wurde, können Sie ihn durch Angabe seines Namens oder nummerischen Bezeichners als Argument für den Befehl `zpool import` importieren. Beispiel:

```
# zpool import tank
```

Wenn mehrere Pools den gleichen Namen besitzen, müssen Sie mithilfe des nummerischen Bezeichners angeben, welcher Pool importiert werden soll. Beispiel:

```
# zpool import
  pool: dozer
    id: 2704475622193776801
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer       ONLINE
          c1t9d0    ONLINE

  pool: dozer
    id: 6223921996155991199
```

```
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer       ONLINE
          c1t8d0    ONLINE
# zpool import dozer
cannot import 'dozer': more than one matching pool
import by numeric ID instead
# zpool import 6223921996155991199
```

Wenn ein Pool-Name mit einem bereits vorhandenen Pool-Namen in Konflikt steht, können Sie den Pool unter einem anderen Namen importieren. Beispiel:

```
# zpool import dozer zeepool
```

Dieser Befehl importiert den exportierten Pool `dozer` unter dem neuen Namen `zeepool`.

Wenn ein Pool nicht ordnungsgemäß exportiert wurde, benötigt ZFS das `-f`-Flag, um zu verhindern, dass versehentlich ein Pool importiert wird, der noch von einem anderen System benutzt wird. Beispiel:

```
# zpool import dozer
cannot import 'dozer': pool may be in use on another system
use '-f' to import anyway
# zpool import -f dozer
```

Pools können mithilfe der Option `-R` in ein anderes Root-Verzeichnis importiert werden. Weitere Informationen zu Speicher-Pools mit alternativem Root-Verzeichnis finden Sie unter „Verwenden von ZFS-Speicher-Pools mit alternativem Root-Verzeichnis“ auf Seite 296.

Wiederherstellen gelöschter ZFS-Speicher-Pools

Gelöschte Speicher-Pools können mit dem Befehl `zpool import -D` wiederhergestellt werden. Beispiel:

```
# zpool destroy tank
# zpool import -D
  pool: tank
    id: 5154272182900538157
 state: ONLINE (DESTROYED)
action: The pool can be imported using its name or numeric identifier.
config:

        tank        ONLINE
          mirror-0  ONLINE
            c1t0d0  ONLINE
            c1t1d0  ONLINE
```

In dieser Ausgabe des Befehls `zpool import` kann der `tank`-Pool aufgrund der folgenden Statusinformationen als gelöscht erkannt werden:

```
state: ONLINE (DESTROYED)
```

Führen Sie zum Wiederherstellen des gelöschten Pools den Befehl `zpool import -D` erneut aus. Beispiel:

```
# zpool import -D tank
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE
          mirror-0  ONLINE
            c1t0d0  ONLINE
            c1t1d0  ONLINE

errors: No known data errors
```

Wenn eines der Datenspeichergeräte im gelöschten Pool fehlerhaft oder nicht verfügbar ist, können Sie den gelöschten Pool unter Umständen mit der Option `-f` trotzdem wiederherstellen. Importieren Sie in einer solchen Situation den im eingeschränkten Zustand befindlichen Pool und versuchen Sie dann, den Geräteausfall zu beheben. Beispiel:

```
# zpool destroy dozer
# zpool import -D
pool: dozer
    id: 13643595538644303788
 state: DEGRADED (DESTROYED)
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
config:

        NAME         STATE     READ WRITE CKSUM
        dozer        DEGRADED     0     0     0
          raidz2-0   DEGRADED     0     0     0
            c2t8d0   ONLINE       0     0     0
            c2t9d0   ONLINE       0     0     0
            c2t10d0  ONLINE       0     0     0
            c2t11d0  UNAVAIL      0    35     1  cannot open
            c2t12d0  ONLINE       0     0     0

errors: No known data errors
# zpool import -Df dozer
# zpool status -x
  pool: dozer
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
```

```
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: scrub completed after 0h0m with 0 errors on Thu Jan 21 15:38:48 2010
config:

        NAME         STATE     READ WRITE CKSUM
        dozer        DEGRADED     0     0     0
          raidz2-0   DEGRADED     0     0     0
            c2t8d0   ONLINE       0     0     0
            c2t9d0   ONLINE       0     0     0
            c2t10d0  ONLINE       0     0     0
            c2t11d0  UNAVAIL      0    37     0  cannot open
            c2t12d0  ONLINE       0     0     0

errors: No known data errors
# zpool online dozer c2t11d0
Bringing device c2t11d0 online
# zpool status -x
all pools are healthy
```

Aktualisieren von ZFS-Speicher-Pools

Wenn in Ihrem System ZFS-Speicher-Pools aus einer früheren Solaris-Version wie z. B. Solaris 10 10/09 vorhanden sind, können Sie diese Pools mit dem Befehl `zpool upgrade` aktualisieren, um die Pool-Funktionen der aktuellen Version nutzen zu können. Darüber hinaus wurde der Befehl `zpool status` so geändert, dass Sie jetzt darauf hingewiesen werden, wenn Pools mit älteren Versionen laufen. Beispiel:

```
# zpool status
  pool: tank
 state: ONLINE
status: The pool is formatted using an older on-disk format.  The pool can
        still be used, but some features are unavailable.
action: Upgrade the pool using 'zpool upgrade'.  Once this is done, the
        pool will no longer be accessible on older software versions.
 scrub: none requested
config:
        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
errors: No known data errors
```

Mithilfe der folgenden Syntax können Sie zusätzliche Informationen zu einer bestimmten Version und unterstützten Releases ermitteln:

```
# zpool upgrade -v
This system is currently running ZFS pool version 22.

The following versions are supported:
```

```
VER  DESCRIPTION
---  --------------------------------------------------------
 1   Initial ZFS version
 2   Ditto blocks (replicated metadata)
 3   Hot spares and double parity RAID-Z
 4   zpool history
 5   Compression using the gzip algorithm
 6   bootfs pool property
 7   Separate intent log devices
 8   Delegated administration
 9   refquota and refreservation properties
 10  Cache devices
 11  Improved scrub performance
 12  Snapshot properties
 13  snapused property
 14  passthrough-x aclinherit
 15  user/group space accounting
 16  stmf property support
 17  Triple-parity RAID-Z
 18  Snapshot user holds
 19  Log device removal
 20  Compression using zle (zero-length encoding)
 21  Reserved
 22  Received properties

For more information on a particular version, including supported releases,
see the ZFS Administration Guide.
```

Anschließend können Sie den Befehl `zpool upgrade` ausführen, um alle Pools zu aktualisieren. Beispiel:

```
# zpool upgrade -a
```

Hinweis – Wenn Sie den Pool auf eine neuere ZFS-Version aktualisieren, ist er auf einem System, auf dem eine ältere ZFS-Version ausgeführt wird, nicht verfügbar.



KAPITEL 5

Installieren und Booten eines Oracle Solaris ZFS-Root-Dateisystems

In diesem Kapitel erfahren Sie, wie ein Oracle Solaris ZFS-Dateisystem installiert und gebootet wird. Darüber hinaus wird die Migration eines UFS-Root-Dateisystems in ein ZFS-Dateisystem mithilfe des Oracle Solaris Live Upgrade behandelt.

Dieses Kapitel enthält die folgenden Abschnitte:

Oracle Solaris 10 9/10 - Versionshinweise.

Stets aktuelle Informationen zur Fehlerbehebung finden Sie unter:

`http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide`

Installieren und Booten eines Oracle Solaris ZFS-Root-Dateisystems (Übersicht)

Ab dem Release Solaris 10 10/08 stehen folgende Methoden zum Installieren und Booten eines ZFS-Root-Dateisystems zur Verfügung:

- Sie können eine Erstinstallation durchführen und dabei ZFS als Root-Dateisystem auswählen.
- Sie können Oracle Solaris Live Upgrade verwenden, um ein UFS-Root-Dateisystem auf ein ZFS-Root-Dateisystem zu migrieren. Außerdem können Sie mit Oracle Solaris Live Upgrade folgende Aufgaben durchführen:
  - Erstellen einer neuen Boot-Umgebung innerhalb eines vorhandenen ZFS-Root-Pools.
  - Erstellen einer neuen Boot-Umgebung in einem neuen ZFS-Root-Pool.
- Sie können ein Oracle Solaris JumpStart-Profil zur automatischen Installation eines Systems mit einem ZFS-Root-Dateisystem verwenden.
- Solaris 10 10/09 ermöglicht Ihnen die automatische Installation eines Systems mit ZFS-Flash-Archiv mithilfe eines JumpStart-Profils.

Nach der Installation eines ZFS-Root-Dateisystems oder der Migration in ein ZFS-Root-Dateisystem auf einem SPARC- oder x86-System startet das System automatisch aus dem ZFS-Root-Dateisystem. Weitere Informationen zum Ändern des Boot-Verhaltens finden Sie unter „Booten aus einem ZFS-Root-Dateisystem“ auf Seite 170.

Leistungsmerkmale für die ZFS-Installation

In diesem Solaris-Release stehen die folgenden Leistungsmerkmale für die ZFS-Installation zur Verfügung:

- Die interaktive Textmodus-Installationsoption von Solaris ermöglicht die Installation eines UFS- oder ZFS-Root-Dateisystems. In diesem Solaris-Release ist UFS weiterhin das Standarddateisystem. Auf die interaktive Textmodus-Installationsoption können Sie wie folgt zugreifen:
  - SPARC: Verwenden Sie die folgende Syntax von der Solaris-Installations-DVD:

    ```
    ok boot cdrom - text
    ```
  - SPARC: Verwenden Sie die folgende Syntax beim Booten über das Netzwerk:

    ```
    ok boot net - text
    ```
  - x86: Wählen Sie die Textmodus-Installationsoption.
- Ein benutzerdefiniertes JumpStart-Profil bietet folgende Funktionen:
  - Sie können ein Profil zum Erstellen eines ZFS-Speicher-Pools anlegen und ein bootfähiges ZFS-Dateisystem benennen.

  - Sie können ein Profil zur Identifizierung eines Flash-Archivs eines ZFS-Root-Pools anlegen.
- Sie können ein UFS-Root-Dateisystem mithilfe des Oracle Solaris Live Upgrade in ein ZFS-Root-Dateisystem migrieren. Die Befehle `lucreate` und `luactivate` wurden um Unterstützung für ZFS-Pools und -Dateisysteme erweitert.
- Sie können bei der Installation zwei Festplatten auswählen und einen gespiegelten Speicher-Pool mit ZFS-Root-Dateisystem (im Folgenden „ZFS-Root-Pool“) einrichten. Alternativ lassen sich auch nach der Installation zusätzliche Festplatten anhängen, um einen gespiegelten ZFS-Root-Pool herzustellen.
- Swap- und Dump-Geräte werden automatisch auf ZFS-Volumes im ZFS-Root-Pool erstellt.

Die folgenden Installationsfunktionen stehen in diesem Release nicht zur Verfügung:

- Die GUI-Installationsfunktion steht derzeit nicht zum Installieren eines ZFS-Root-Dateisystems zur Verfügung.
- Die Oracle Solaris Flash-Installationsfunktion zum Installieren eines ZFS-Root-Dateisystems ist nicht verfügbar, wenn die Option "Flash-Installation" aus der Erstinstallationsoption gewählt wird. Sie können jedoch ein JumpStart-Profil zur Identifizierung eines Flash-Archivs eines ZFS-Root-Pools erstellen. Weitere Informationen finden Sie unter „Installieren eines ZFS-Root-Dateisystems (Oracle Solaris Flash-Archiv-Installation)“ auf Seite 137.
- Ein Upgrade des UFS-Root-Dateisystems zu einem ZFS-Root-Dateisystem ist nicht mithilfe des Standard-Upgrade-Programms möglich.

Oracle Solaris-Installation und Oracle Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung

Vergewissern Sie sich vor dem Versuch, ein ZFS-Root-Dateisystem auf einem System zu installieren oder ein UFS- in ein ZFS-Root-Dateisystem zu migrieren, dass folgende Voraussetzungen erfüllt sind:

Voraussetzungen für die Oracle Solaris-Version

Sie haben folgende Möglichkeiten, ein ZFS-Root-Dateisystem zu installieren und zu booten oder in ein ZFS-Root-Dateisystem zu migrieren:

- Installieren eines ZFS-Root-Dateisystems – ab Solaris 10 10/08 verfügbar.
- Migrieren aus einem UFS-Root-Dateisystem in ein ZFS-Root-Dateisystem mit Oracle Solaris Live Upgrade. – Sie müssen Solaris 10 10/08 oder eine höhere Version installiert oder auf Solaris 10 10/08 oder eine höhere Version aktualisiert haben.

Allgemeine Voraussetzungen für einen ZFS-Speicher-Pool

In den folgenden Abschnitte werden der Speicherplatz für ZFS-Root-Pools und Konfigurationsvoraussetzungen beschrieben.

Erforderliche Festplattenkapazität für ZFS-Speicher-Pools

Ein ZFS-Root-Dateisystem benötigt mehr Speicherplatz im Pool als ein UFS-Root-Dateisystem, da Swap- und Dump-Geräte in einer ZFS-Root-Umgebung separate Speichergeräte sein müssen. In UFS-Root-Dateisystemen sind Swap- und Dump-Geräte standardmäßig dasselbe Gerät.

Wenn auf einem System ein ZFS-Root-Dateisystem installiert oder ein Upgrade darauf vorgenommen wird, hängen die Größe des Swap-Bereichs und des Dump-Geräts von der physischen Speicherkapazität ab. Der mindestens erforderliche Speicherplatz im Pool für ein boot-fähiges ZFS-Root-Dateisystem richtet sich nach der Größe des physischen Speichers, dem freien Speicherplatz auf der Festplatte und der Anzahl der zu erstellenden Boot-Umgebungen (BUs).

Beachten Sie die erforderliche Festplattenkapazität für ZFS-Speicher-Pools:

- Für die Installation eines ZFS-Root-Dateisystems sind mindestens 768 MB Arbeitsspeicher erforderlich.
- Für die optimale Leistung des gesamten ZFS-Dateisystems wird 1 GB empfohlen.
- Es werden mindestens 16 GB Festplattenkapazität empfohlen. Der Festplattenspeicher wird wie folgt belegt:
  - **Swap-Bereich und Dump-Gerät** – Die Solaris-Installationsprogramme erstellen Swap- und Dump-Volumes in den folgenden Standardgrößen:
    - **Erstinstallation von Solaris** – Die Größe des Standard-Swap-Volume ist in der neuen ZFS-BU halb so groß wie der physische Speicher, im Allgemeinen zwischen 512 MB und 2 GB. Die Größe des Swap-Volume kann während einer Erstinstallation angepasst werden.
    - Die Dump-Volume-Größe wird vom Kernel basierend auf den `dumpadm`-Informationen und der Größe des physikalischen Speichers berechnet. Die Größe des Dump-Volume kann während einer Erstinstallation angepasst werden.
    - **Oracle Solaris Live Upgrade** – Wenn ein UFS-Root-Dateisystem auf ein ZFS-Root-Dateisystem umgestellt wird, wird die Größe des Standard-Swap-Volume für die ZFS-Boot-Umgebung (ZFS-BU) als Größe des Swap-Geräts der UFS-BU berechnet. Bei der Berechnung der Größe des Standard-Swap-Volume werden die Größen aller Swap-Geräte in der UFS-BU addiert, und es wird ein ZFS-Volume der entsprechenden Größe in der ZFS-BU erstellt. Wenn keine Swap-Geräte in der UFS-BU definiert sind, wird die Größe des Standard-Swap-Volume auf 512 MB gesetzt.

- Die Größe des Standard-Dump-Volume ist in der ZFS-BU halb so groß wie der physische Speicher, zwischen 512 MB und 2 GB.

  Sie können die Größen der Swap- und Dump-Volumes ändern, sofern die neuen Größen den Betrieb des Systems unterstützen. Weitere Informationen finden Sie unter „Anpassen der Größe von ZFS-Swap- und Dump-Geräten" auf Seite 167.

- **Boot-Umgebung (BU)** – Zusätzlich zu dem für neue Swap- und Dump-Geräte erforderlichen oder nachträglich geänderten Speicherplatz werden für eine ZFS-BU bei Migration von einer UFS-BU ungefähr 6 GB benötigt. Für ZFS-BUs, die aus anderen ZFS-BUs geklont werden, ist kein zusätzlicher Speicherplatz erforderlich. Beachten Sie jedoch, dass die BU-Größe durch die Installation von Patches zunimmt. Alle ZFS-BUs in demselben Root-Pool greifen auf dieselben Swap- und Dump-Geräte zu.
- **Komponenten des Solaris-Betriebssystems** – Alle Unterverzeichnisse des Root-Dateisystems, die zum Betriebssystem-Image gehören, ausgenommen `/var` müssen im selben Dataset wie das Root-Dateisystem enthalten sein. Außerdem müssen alle Komponenten des Solaris-Betriebssystems im Root-Pool enthalten sein, mit Ausnahme der Swap- und Dump-Geräte.

  Eine weitere Beschränkung ist, dass das Verzeichnis bzw. Dataset `/var` ein einzelnes Dataset sein muss. Sie können kein untergeordnetes `/var`-Dataset wie beispielsweise `/var/tmp` erstellen, wenn Sie Oracle Solaris Live Upgrade verwenden möchten, um eine ZFS-BU zu migrieren oder zu patchen oder ein ZFS-Flash-Archiv dieses Pools zu erstellen.

Beispielsweise könnte ein System mit 12 GB Festplattenkapazität zu klein für eine boot-fähige ZFS-Umgebung sein, da 2 GB je Swap- und Dump-Gerät und rund 6 GB für die ZFS-BU benötigt werden, die aus einer UFS-BU migriert wird.

Voraussetzungen für die Konfiguration des ZFS-Speicher-Pools

Machen Sie sich mit folgenden Voraussetzungen bezüglich der Konfiguration von ZFS-Speicher-Pools vertraut:

- Der als Root-Pool bestimmte Pool muss ein SMI-Label haben. Diese Anforderung wird erfüllt, wenn der Pool mithilfe von Festplattenbereichen erstellt wird.
- Der Pool muss entweder auf einem Festplattenbereich oder auf gespiegelten Festplattenbereichen vorhanden sein. Wenn Sie versuchen, eine nicht unterstützte Pool-Konfiguration bei einer Oracle Solaris Live Upgrade-Migration zu verwenden, wird eine Meldung wie die folgende angezeigt:

  ```
  ERROR: ZFS pool name does not support boot environments
  ```

  Weitere Informationen zu unterstützten Konfigurationen von ZFS-Root-Pools finden Sie unter „Erstellen eines ZFS-Root-Pools" auf Seite 73.

- x86: Die Festplatte muss eine Solaris-`fdisk`-Partition enthalten. Eine Solaris-`fdisk`-Partition wird bei der Installation eines x86-Systems automatisch installiert. Weitere Informationen zu Solaris-`fdisk`-Partitionen finden Sie in „Guidelines for Creating an fdisk Partition“ in *System Administration Guide: Devices and File Systems*.
- Datenträger, die für das Booten in einem ZFS-Root-Pool bestimmt sind, dürfen auf SPARC- sowie auf x86-Systemen nicht mehr als 1 TB umfassen.
- Auf dem Root-Pool kann die Komprimierung aktiviert werden, allerdings erst nach Installation des Root-Pools. Während der Installation eines Root-Pools kann die Komprimierung nicht aktiviert werden. Der Komprimierungs-Algorithmus `gzip` wird auf Root-Pools nicht unterstützt.
- Benennen Sie den Root-Pool nicht um, nachdem dieser in einer Erstinstallation erstellt wurde oder nachdem eine Solaris Live Upgrade-Migration auf ein ZFS-Root-Dateisystem durchgeführt wurde. Das Umbenennen des Root-Pools kann dazu führen, dass das System nicht gebootet werden kann.

Installieren eines ZFS-Root-Dateisystems (Erstinstallation)

In diesem Solaris-Release können Sie eine Erstinstallation durchführen und mit der interaktiven Textmodus-Installationsoption von Solaris einen ZFS-Speicher-Pool erstellen, das ein boot-fähiges ZFS-Root-Dateisystem enthält. Wenn Sie einen bereits vorhandenen ZFS-Speicher-Pool für das ZFS-Root-Dateisystem verwenden möchten, müssen Sie das vorhandene UFS-Root-Dateisystem mit Oracle Solaris Live Upgrade in ein ZFS-Root-Dateisystem in einem vorhandenen ZFS-Speicher-Pool migrieren. Weitere Informationen finden Sie unter „Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem (Oracle Solaris Live Upgrade)“ auf Seite 144.

Informationen zum Konfigurieren von Zonen und zum Patchen oder Aktualisieren des Systems nach der Erstinstallation eines ZFS-Root-Dateisystems finden Sie unter „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 150 oder „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (ab Solaris 10 5/09)“ auf Seite 156.

Sollten bereits ZFS-Speicher-Pools auf dem System vorhanden sein, werden sie durch die folgende Meldung bestätigt. Diese Pools bleiben jedoch unverändert, es sei denn, Sie wählen die Festplatten in den vorhandenen Pools aus, um den neuen Speicher-Pool zu erstellen.

```
There are existing ZFS pools available on this system.  However, they can only be upgraded
using the Live Upgrade tools.  The following screens will only allow you to install a ZFS root system,
not upgrade one.
```



Achtung – Vorhandene Pools werden überschrieben, falls einige ihrer Datenträger für den neuen Pool ausgewählt werden.

Bevor Sie mit der Erstinstallation für die Erstellung eines ZFS-Speicher-Pools beginnen, lesen Sie den Abschnitt „Oracle Solaris-Installation und Oracle Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung" auf Seite 127.

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems

Der interaktive, textgestützte Solaris-Installationsprozess ist im Wesentlichen derselbe wie in der vorherigen Solaris-Versionen, mit der Ausnahme, dass Sie gefragt werden, ob ein UFS- oder ZFS-Root-Dateisystem erstellt werden soll. UFS ist auch in diesem Release weiterhin das Standarddateisystem. Wenn Sie ein ZFS-Root-Dateisystem wählen, werden Sie aufgefordert, einen ZFS-Speicher-Pool zu erstellen. Es folgen die Schritte zur Installation eines ZFS-Root-Dateisystems:

1. Wählen Sie die interaktive Solaris-Installationsmethode, da die Solaris Flash-Installation zum Erstellen eines boot-fähigen ZFS-Root-Dateisystems nicht verfügbar ist. Sie können jedoch ein ZFS-Flash-Archiv zur Verwendung während einer JumpStart-Installation erstellen. Weitere Informationen finden Sie unter „Installieren eines ZFS-Root-Dateisystems (Oracle Solaris Flash-Archiv-Installation)" auf Seite 137.

   Ab der Solaris-Version 10 10/08 können Sie ein UFS-Root-Dateisystem in ein ZFS-Root-Dateisystem migrieren. Hierfür muss mindestens Solaris 10 10/08 installiert sein. Weitere Informationen zur Migration in ein ZFS-Root-Dateisystem finden Sie unter „Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem (Oracle Solaris Live Upgrade)" auf Seite 144.

2. Wenn Sie ein ZFS-Root-Dateisystem erstellen möchten, wählen Sie die ZFS-Option. Beispiel:

   ```
   Choose Filesystem Type

     Select the filesystem to use for your Solaris installation


                [ ] UFS
                [X] ZFS
   ```

3. Nach der Auswahl der zu installierenden Software werden Sie zur Auswahl der Festplatten aufgefordert, auf denen der ZFS-Speicher-Pool installiert werden soll. Dieser Bildschirm sieht ähnlich aus wie in vorherigen Solaris-Versionen.

   ```
   Select Disks

     On this screen you must select the disks for installing Solaris software.
     Start by looking at the Suggested Minimum field; this value is the
     approximate space needed to install the software you've selected. For ZFS,
     multiple disks will be configured as mirrors, so the disk you choose, or the
     slice within the disk must exceed the Suggested Minimum value.
   ```

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

```
  NOTE: ** denotes current boot disk

  Disk Device                                              Available Space
  =============================================================================
  [X]    c1t0d0                                           69994 MB  (F4 to edit)
  [ ]    c1t1d0                                           69994 MB
  [-]    c1t2d0                                               0 MB
  [-]    c1t3d0                                               0 MB

                                  Maximum Root Size:  69994 MB
                                  Suggested Minimum:   8279 MB
```

Sie können die Festplatte(n) auswählen, auf der/denen ein ZFS-Root-Pool installiert werden soll. Wenn Sie zwei Festplatten auswählen, wird für den Root-Pool eine aus zwei Platten bestehende Konfiguration mit Datenspiegelung ausgewählt. Optimal ist ein gespiegelter Pool mit zwei oder drei Festplatten. Wenn Sie über acht Datenträger verfügen und alle auswählen, werden diese acht Festplatten als große Datenspiegelung für den Root-Pool verwendet. Diese Konfiguration ist nicht optimal. Alternativ können Sie einen gespiegelten Root-Pool nach der Erstinstallation erstellen. RAID-Z-Konfigurationen werden für den Root-Pool nicht unterstützt. Weitere Informationen zur Konfiguration von ZFS-Speicher-Pools finden Sie unter „Replikationsfunktionen eines ZFS-Speicher-Pools“ auf Seite 69.

4. Um zwei Festplatten zur Erstellung eines gespiegelten Root-Pools auszuwählen, wählen Sie die zweite Festplatte mit dem Cursor aus. Im folgenden Beispiel werden sowohl `c1t1d1` als auch `c0t2d0` als Root-Pool-Festplatten ausgewählt. Beide Festplatten müssen ein SMI-Label und den Bereich 0 haben. Falls die Festplatten kein SMI-Label haben oder Bereiche enthalten, verlassen Sie das Installationsprogramm und verwenden das Dienstprogramm `format` zur Umbenennung und Neupartitionierung der Festplatten. Anschließend starten Sie das Installationsprogramm erneut.

```
Select Disks

  On this screen you must select the disks for installing Solaris software.
  Start by looking at the Suggested Minimum field; this value is the
  approximate space needed to install the software you've selected. For ZFS,
  multiple disks will be configured as mirrors, so the disk you choose, or the
  slice within the disk must exceed the Suggested Minimum value.
  NOTE: ** denotes current boot disk

  Disk Device                                              Available Space
  =============================================================================
  [X]    c1t0d0                                           69994 MB
  [X]    c1t1d0                                           69994 MB  (F4 to edit)
  [-]    c1t2d0                                               0 MB
  [-]    c1t3d0                                               0 MB

                                  Maximum Root Size:  69994 MB
                                  Suggested Minimum:   8279 MB
```

Wenn in der Spalte "Verfügbarer Speicherplatz" 0 MB angezeigt werden, bedeutet dies in der Regel, dass die Festplatte ein EFI-Label hat. Wenn Sie eine Festplatte mit einem

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

EFI-Label verwenden möchten, verlassen Sie das Installationsprogramm, versehen die Festplatte mit einem neuen SMI-Label, indem Sie den Befehl `format -e` verwenden, und starten anschließend das Installationsprogramm erneut.

Wenn Sie während der Installation keinen gespiegelten Root-Pool erstellen, können Sie dies nach der Installation problemlos nachholen. Weitere Informationen finden Sie unter „So erstellen Sie einen gespiegelten Root-Pool (nach der Installation)" auf Seite 136.

5. Nach der Auswahl eines bzw. mehrerer Datenträger für den ZFS-Speicher-Pool wird ein Bildschirm wie der folgende angezeigt:

```
Configure ZFS Settings

  Specify the name of the pool to be created from the disk(s) you have chosen.
  Also specify the name of the dataset to be created within the pool that is
  to be used as the root directory for the filesystem.

              ZFS Pool Name: rpool
      ZFS Root Dataset Name: s10s_u9wos_08
      ZFS Pool Size (in MB): 69995
  Size of Swap Area (in MB): 2048
  Size of Dump Area (in MB): 1536
        (Pool size must be between 6231 MB and 69995 MB)

                         [X] Keep / and /var combined
                         [ ] Put /var on a separate dataset
```

   Auf diesem Bildschirm können Sie den Namen des ZFS-Pools, den Dataset-Namen, die Pool-Größe sowie die Größe der Swap- und Dump-Geräte ändern. Verschieben Sie hierzu den Cursor in den Einträgen und ersetzen Sie die Standardwerte durch die gewünschten Werte. Anderenfalls können Sie die Standardwerte übernehmen. Außerdem können Sie das Verfahren ändern, mit dem das Dateisystem `/var` erstellt und eingehängt wird.

   In diesem Beispiel wird der Name des Root-Datasets in `zfsBE` abgeändert.

```
              ZFS Pool Name: rpool
      ZFS Root Dataset Name: zfsBE
      ZFS Pool Size (in MB): 69995
  Size of Swap Area (in MB): 2048
  Size of Dump Area (in MB): 1536
        (Pool size must be between 6231 MB and 69995 MB)

                         [X] Keep / and /var combined
                         [ ] Put /var on a separate dataset
```

6. Dieser letzte Installationsbildschirm bietet die Möglichkeit, das Installationsprofil zu ändern. Beispiel:

```
Profile

  The information shown below is your profile for installing Solaris software.
  It reflects the choices you've made on previous screens.
```

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

```
============================================================================
                Installation Option: Initial
                        Boot Device: c1t0d0
              Root File System Type: ZFS
                    Client Services: None

                            Regions: North America
                      System Locale: C ( C )

                           Software: Solaris 10, Entire Distribution
                          Pool Name: rpool
              Boot Environment Name: zfsBE
                          Pool Size: 69995 MB
                    Devices in Pool: c1t0d0
                                     c1t1d0
```

7. Überprüfen Sie nach abgeschlossener Installation die Informationen zum erstellten ZFS-Speicher-Pool und Dateisystem. Beispiel:

```
# zpool status
  pool: rpool
 state: ONLINE
 scrub: none requested
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c1t0d0s0  ONLINE       0     0     0
            c1t1d0s0  ONLINE       0     0     0

errors: No known data errors
# zfs list
NAME                USED  AVAIL  REFER  MOUNTPOINT
rpool              8.03G  58.9G    96K  /rpool
rpool/ROOT         4.47G  58.9G    21K  legacy
rpool/ROOT/zfsBE   4.47G  58.9G  4.47G  /
rpool/dump         1.50G  58.9G  1.50G  -
rpool/export         44K  58.9G    23K  /export
rpool/export/home    21K  58.9G    21K  /export/home
rpool/swap         2.06G  61.0G    16K  -
```

In der Beispielausgabe von `zfs list` sind die Root-Pool-Komponenten wie z. B. das Verzeichnis `rpool/ROOT` aufgeführt, auf das standardmäßig nicht zugegriffen werden kann.

8. Um eine weitere ZFS-Boot-Umgebung (BU) im selben Speicher-Pool zu erstellen, verwenden Sie den Befehl `lucreate`. Im folgenden Beispiel wird eine neue BU namens `zfs2BE` erstellt. Die aktuelle BU wird mit `zfsBE` benannt, wie in der Ausgabe von `zfs list` gezeigt wird. Die aktuelle BU wird jedoch erst nach der Erstellung der neuen BU in der Ausgabe von `lustatus` bestätigt.

```
# lustatus
ERROR: No boot environments are configured on this system
ERROR: cannot determine list of all boot environment names
```

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

Zum Erstellen einer neuen ZFS-BU in demselben Pool verwenden Sie folgende Syntax:

```
# lucreate -n zfs2BE
INFORMATION: The current boot environment is not named - assigning name <zfsBE>.
Current boot environment is named <zfsBE>.
Creating initial configuration for primary boot environment <zfsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <zfsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <zfsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs2BE>.
Source boot environment is <zfsBE>.
Creating boot environment <zfs2BE>.
Cloning file systems from boot environment <zfsBE> to create boot environment <zfs2BE>.
Creating snapshot for <rpool/ROOT/zfsBE> on <rpool/ROOT/zfsBE@zfs2BE>.
Creating clone for <rpool/ROOT/zfsBE@zfs2BE> on <rpool/ROOT/zfs2BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs2BE>.
Population of boot environment <zfs2BE> successful.
Creation of boot environment <zfs2BE> successful.
```

Beim Erstellen einer ZFS-BU innerhalb desselben Pools kommen die Klon- und Snapshot-Funktionen von ZFS zum Einsatz und die BU wird sofort erstellt. Weitere Informationen zur ZFS-Root-Migration mithilfe von Oracle Solaris Live Upgrade finden Sie unter „Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem (Oracle Solaris Live Upgrade)“ auf Seite 144.

9. Überprüfen Sie anschließend die neuen Boot-Umgebungen. Beispiel:

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      yes    yes       no     -
zfs2BE                     yes      no     no        yes    -
# zfs list
NAME                      USED  AVAIL  REFER  MOUNTPOINT
rpool                    8.03G  58.9G    97K  /rpool
rpool/ROOT               4.47G  58.9G    21K  legacy
rpool/ROOT/zfs2BE         116K  58.9G  4.47G  /
rpool/ROOT/zfsBE         4.47G  58.9G  4.47G  /
rpool/ROOT/zfsBE@zfs2BE  75.5K      -  4.47G  -
rpool/dump               1.50G  58.9G  1.50G  -
rpool/export               44K  58.9G    23K  /export
rpool/export/home          21K  58.9G    21K  /export/home
rpool/swap               2.06G  61.0G    16K  -
```

10. Zum Booten aus einer alternativen BU verwenden Sie den Befehl `luactivate`. Nachdem Sie die BU auf einem SPARC-basierten System aktiviert haben, können Sie mithilfe des Befehls `boot - L` verfügbare BUs identifizieren, wenn das Boot-Gerät einen ZFS-Speicher-Pool enthält. Beim Booten eines x86-Systems ermitteln Sie die zu bootende BU über das GRUB-Menü.

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

Geben Sie also beispielsweise bei einem SPARC-System den Befehl `boot -L` ein, um eine Liste der verfügbaren BUs anzuzeigen. Zum Booten aus der neuen BU `zfs2BE` wählen Sie Option 2. Geben Sie dann den angezeigten Befehl `boot -Z` ein.

```
ok boot -L
Executing last command: boot -L
Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0  File and args: -L
1 zfsBE
2 zfs2BE
Select environment to boot: [ 1 - 2 ]: 2

To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/zfs2BE
ok boot -Z rpool/ROOT/zfs2BE
```

Weitere Informationen zum Booten eines ZFS-Dateisystems können Sie dem Abschnitt „Booten aus einem ZFS-Root-Dateisystem“ auf Seite 170 entnehmen.

▼ So erstellen Sie einen gespiegelten Root-Pool (nach der Installation)

Wenn Sie während der Installation keinen gespiegelten ZFS-Root-Pool erstellen, können Sie dies nach der Installation problemlos nachholen.

Informationen zum Ersetzen einer Festplatte in einem ZFS-Root-Pool finden Sie unter „So ersetzen Sie eine Festplatte im ZFS-Root-Pool“ auf Seite 177.

1 **Zeigen Sie den aktuellen Root-Pool-Status an.**

```
# zpool status rpool
  pool: rpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        rpool       ONLINE       0     0     0
          c1t0d0s0  ONLINE       0     0     0

errors: No known data errors
```

2 **Binden Sie eine zweite Festplatte ein, um einen gespiegelten Root-Pool zu konfigurieren.**

```
# zpool attach rpool c1t0d0s0 c1t1d0s0
Please be sure to invoke installboot(1M) to make 'c1t1d0s0' bootable.
Make sure to wait until resilver is done before rebooting.
```

3 **Zeigen Sie den Root-Pool-Status an, um zu bestätigen, dass das Resilvering abgeschlossen ist.**

```
# zpool status rpool
  pool: rpool
 state: ONLINE
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress for 0h1m, 24.26% done, 0h3m to go
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c1t0d0s0  ONLINE       0     0     0
            c1t1d0s0  ONLINE       0     0     0  3.18G resilvered

errors: No known data errors
```

In der obigen Ausgabe ist der Resilvering-Prozess nicht abgeschlossen. Das Resilvering ist abgeschlossen, wenn eine Meldung wie die folgende angezeigt wird:

```
scrub: resilver completed after 0h10m with 0 errors on Thu Mar 11 11:27:22 2010
```

4 **Wenden Sie Boot-Blöcke auf die zweite Festplatte an, wenn das Resilvering abgeschlossen ist.**

```
sparc# installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c1t1d0s0
```

```
x86# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c1t1d0s0
```

5 **Überprüfen Sie, ob Sie von der zweiten Festplatte booten können.**

6 **Richten Sie das System zum automatischen Booten von der neuen Festplatte ein, indem Sie den Befehl `eeprom` oder den Befehl `setenv` vom SPARC-Boot-PROM verwenden. Sie können aber auch das PC-BIOS neu konfigurieren.**

Installieren eines ZFS-Root-Dateisystems (Oracle Solaris Flash-Archiv-Installation)

In Solaris 10 10/09 kann ein Flash-Archiv auf einem System erstellt werden, auf dem ein UFS-Root-Dateisystem oder ein ZFS-Root-Dateisystem ausgeführt wird. Ein Flash-Archiv eines ZFS-Root-Pools enthält die gesamte Pool-Hierarchie außer Swap- und Dump-Volumes und ausgeschlossene Datensätze. Die Swap- und Dump-Volumes werden bei der Installation des Flash-Archivs erstellt. Sie können bei der Flash-Archiv-Installation wie folgt vorgehen:

- Erstellen Sie ein Flash-Archiv, das zur Installation und zum Starten eines Systems mit einem ZFS-Root-Dateisystem verwendet werden kann.

- Führen Sie eine JumpStart-Installation eines Systems unter Verwendung eines ZFS-Flash-Archivs aus. Durch die Erstellung eines ZFS-Flash-Archivs wird ein ganzer Root Pool geklont, nicht nur einzelne Boot-Umgebungen. Einzelne Datensätze innerhalb des Pools können mit der Option `D` der Befehle `flarcreate` und `-flar` ausgeschlossen werden.

Beachten Sie folgende Einschränkungen, bevor Sie ein System mit einem ZFS-Flash-Archiv installieren:

- Nur die JumpStart-Installation eines ZFS-Flash-Archivs wird unterstützt. Sie können die interaktive Installationsoption eines Flash-Archivs nicht zur Installation eines Systems mit einem ZFS-Root-Dateisystem verwenden. Des Weiteren ist es nicht möglich, ein Flash-Archiv zur Installation einer ZFS-BU mit Oracle Solaris Live Upgrade zu verwenden.
- Sie können ein Flash-Archiv nur auf einem System installieren, das dieselbe Architektur wie das System besitzt, auf dem Sie das ZFS-Flash-Archiv erstellt haben. Beispielsweise kann ein auf einem `sun4u`-System erstelltes Archiv nicht auf einem `sun4v` -System installiert werden.
- Es wird nur die vollständige Erstinstallation eines ZFS-Flash-Archivs unterstützt. Sie können weder ein anderes Flash-Archiv eines ZFS-Root-Dateisystems noch ein Hybrid-UFS/ZFS-Archiv installieren.
- Vorhandene UFS-Flash-Archive können weiterhin nur zur Installation eines UFS-Root-Dateisystems verwendet werden. Das ZFS-Flash-Archiv kann nur zur Installation eines ZFS-Root-Dateisystems verwendet werden.
- Auch wenn der gesamte Root-Pool, ausdrücklich ausgeschlossene Datasets ausgenommen, archiviert und installiert wird, ist nur die ZFS-BU, die bei Erstellung des Archivs gebootet wird, nach Installation des Flash-Archivs verwendbar. Jedoch können mit der Befehlsoption `flarcreate` oder `flar - R` *rootdir* archivierte Pools zur Archivierung eines anderen als des aktuell gebooteten Root-Pools verwendet werden.
- Ein mit einem Flash-Archiv erstellter Name eines ZFS-Root-Pools muss mit dem Namen des Master-Root-Pools übereinstimmen. Der Name des Root-Pools, der zur Erstellung des Flash-Archivs verwendet wird, wird dem neu erstellten Pool zugewiesen. Der Pool-Name kann nicht verändert werden.
- Die Befehlsoptionen `flarcreate` und `flar` zum Ein- und Ausschließen einzelner Dateien werden in einem ZFS-Flash-Archiv nicht unterstützt. Sie können nur komplette Datasets aus einem ZFS-Flash-Archiv ausschließen.
- Der Befehl `flar info` wird für ein ZFS-Flash-Archiv nicht unterstützt. Beispiel:

```
# flar info -l zfs10u8flar
ERROR: archive content listing not supported for zfs archives.
```

Nachdem ein Master-System mit Solaris 10 10/09 oder einer höheren Version installiert oder aktualisiert wurde, können Sie ein ZFS-Flash-Archiv erstellen, um ein Zielsystem zu installieren. Dieser Vorgang läuft im Wesentlichen wie folgt ab:

- Installieren Sie Solaris 10 10/09 oder eine höhere Version auf dem Master-System, oder verwenden Sie Solaris 10 10/09 oder eine höhere Version, um das Master-System zu aktualisieren. Nehmen Sie ggf. benutzerdefinierte Einstellungen vor.

- Erstellen Sie das ZFS-Flash-Archiv mit dem Befehl `flarcreate` auf dem Master-System. Alle Datasets im Root-Pool außer Swap- und Dump-Volumes werden im ZFS-Flash-Archiv beinhaltet.
- Erstellen Sie ein JumpStart-Profil, um die Flash-Archiv-Information auf dem Installationsserver zu beinhalten.
- Installieren Sie das ZFS-Flash-Archiv auf dem Zielsystem.

Die folgenden Archivierungsoptionen werden zur Installation eines ZFS-Root-Pools mit einem Flash-Archiv unterstützt:

- Verwenden Sie den Befehl `flarcreate` oder `flar`, um ein Flash-Archiv aus dem angegebenen ZFS-Root-Pool zu erstellen. Falls kein ZFS-Root-Pool angegeben ist, wird ein Flash-Archiv aus dem Standard-Root-Pool erstellt.
- Verwenden Sie `flarcreate -D` *dataset*, um die angegebenen Datasets aus dem Flash-Archiv auszuschließen. Diese Option kann mehrmals zum Ausschließen mehrerer Datasets verwendet werden.

Nach Installation eines ZFS-Flash-Archivs wird das System wie folgt konfiguriert:

- Die komplette auf dem System, auf dem das Flash-Archiv erstellt wurde, vorhandene Dataset-Hierarchie wird auf dem Zielsystem neu erstellt, ausgenommen jegliche Datasets, die bei der Erstellung des Archivs ausgeschlossen wurden. Die Swap- und Dump-Volumes werden nicht in das Flash-Archiv eingeschlossen.
- Der Root-Pool hat den gleichen Namen wie der Pool, aus dem das Archiv erstellt wurde.
- Die zum Zeitpunkt der Erstellung des Flash-Archivs aktive Boot-Umgebung ist die aktive und standardmäßige BU auf den verwendeten Systemen.

BEISPIEL 5–2 Installieren eines ZFS-Flash-Archivs auf einem System

Nachdem Sie das Master-System mit Solaris 10 10/09 oder einer höheren Version installiert oder aktualisiert haben, erstellen Sie ein Flash-Archiv des ZFS-Root-Pools. Beispiel:

```
# flarcreate -n zfsBE zfs10upflar
Full Flash
Checking integrity...
Integrity OK.
Running precreation scripts...
Precreation scripts done.
Determining the size of the archive...
The archive will be approximately 4.94GB.
Creating the archive...
Archive creation complete.
Running postcreation scripts...
Postcreation scripts done.

Running pre-exit scripts...
Pre-exit scripts done.
```

Erstellen Sie auf dem System, das als Installationsserver dienen soll, ein JumpStart-Profil. Gehen Sie dabei so wie bei der normalen Installation eines Systems vor. Das folgende Profil wird beispielsweise für die Installation des `zfs10upflar`-Archivs verwendet.

```
install_type flash_install
archive_location nfs system:/export/jump/zfs10upflar
partitioning explicit
pool rpool auto auto auto mirror c0t1d0s0 c0t0d0s0
```

Installieren eines ZFS-Root-Dateisystems (Oracle Solaris JumpStart-Installation)

Sie können eine JumpStart-Profil erstellen, um ein ZFS- oder UFS-Root-Dateisystem zu installieren.

ZFS-spezifische Profile müssen das neue Schlüsselwort `pool` enthalten. Mit dem Schlüsselwort `pool` wird ein neuer Root-Pool installiert. Dabei wird standardmäßig eine neue Boot-Umgebung erstellt. Sie haben die Möglichkeit, einen Namen für die Boot-Umgebung anzugeben und ein separates `/var` -Dataset zu erstellen. Hierzu dienen die Schlüsselwörter `bootenv installbe` und die Optionen `bename` und `dataset`.

Allgemeine Informationen zu den JumpStart-Funktionen finden Sie im *Oracle Solaris 10 9/10 Installationshandbuch: Benutzerdefinierte JumpStart-Installation und komplexe Installationsszenarien*.

Informationen zum Konfigurieren von Zonen und zum Patchen oder Aktualisieren des Systems nach der JumpStart-Installation eines ZFS-Root-Dateisystems finden Sie unter „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 150 oder „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (ab Solaris 10 5/09)“ auf Seite 156.

JumpStart-Schlüsselwörter für ZFS

Die folgenden Schlüsselwörter sind in ZFS-spezifischen Profilen zulässig:

`auto` Legt automatisch die Größe der Bereiche für Pool, Swap-Volume oder Dump-Volume fest. Die Größe der Festplatte wird auf Einhaltung der Mindestgröße überprüft. Wenn die Mindestgröße verfügbar ist, wird die im Rahmen der gegebenen Einschränkungen (Größe der Festplatten, reservierte Bereiche usw.) maximale Pool-Größe zugewiesen.

Wenn Sie beispielsweise `c0t0d0s0` und eines der Schlüsselwörter `all` oder `auto` angeben, wird der Root-Pool-Bereich so groß wie möglich erstellt. Sie können aber auch eine bestimmte Größe für den Bereich, das Swap-Volume oder das Dump-Volume angeben.

Im Zusammenhang mit einem ZFS-Root-Pool verhält sich das Schlüsselwort `auto` wie das Schlüsselwort `all`, da Pools keine ungenutzte Festplattenkapazität haben.

`bootenv` Mit diesem Schlüsselwort werden die Merkmale der Boot-Umgebung festgelegt.

Zum Erstellen einer boot-fähigen ZFS-Root-Umgebung verwenden Sie das Schlüsselwort `bootenv` mit der folgenden Syntax:

`bootenv installbe bename` *BU-Name* [*dataset Einhängepunkt*]

`installbe` Erstellt eine neue, durch die Option `bename` und den Eintrag *BU-Name* näher bezeichnete BU und installiert sie.

`bename` *BU-Name* Gibt den *BU-Namen* für die Installation an.

Wenn Sie `bename` nicht zusammen mit dem Schlüsselwort `pool` angeben, wird eine Standard-BU erstellt.

`dataset` *Einhängepunkt* Das optionale Schlüsselwort `dataset` ermöglicht es, ein vom Root-Dataset separates `/var`-Dataset anzugeben. Der Wert für *Einhängepunkt* ist derzeit auf `/var` beschränkt. So würde beispielsweise eine `bootenv`-Syntaxzeile für ein separates `/var`-Dataset wie folgt aussehen:

```
bootenv installbe bename zfsroot dataset /var
```

`pool` Definiert den neuen, zu erstellenden Root-Pool. Die folgende Schlüsselwortsyntax muss verwendet werden:

`pool` *poolname poolsize swapsize dumpsize vdevlist*

Pool-Name Gibt den Namen des zu erstellenden Pools an. Der Pool wird mit der angegebenen *Größe* und den angegebenen physischen Geräten (*vdevs*) erstellt. Geben Sie mit dem Wert `poolname` nicht den Namen eines vorhandenen Pools an, sonst wird der vorhandene Pool überschrieben.

Pool-Größe Gibt die Größe des zu erstellenden Pools an. Der Wert kann `auto` oder `existing` sein. Mit dem Wert `auto` wird die im Rahmen der gegebenen Einschränkungen (Größe der Festplatten, reservierte Bereiche usw.) maximale Pool-Größe zugewiesen. Der Wert `existing` bewirkt, dass die Grenzen vorhandener Bereiche mit diesem Namen beibehalten und nicht überschrieben werden. Sofern nicht `g` (GB) angegeben wird, gilt für die Größenangabe die Einheit MB.

Swap-Größe Gibt die Größe des zu erstellenden Swap-Volumes an. Der Wert auto bewirkt, dass die Standard-Swap-Größe verwendet wird. Sie können eine Größe angeben, indem Sie einen *size*-Wert verwenden. Sofern nicht g (GB) angegeben wird, gilt für die Größenangabe die Einheit MB.

Dump-Größe Gibt die Größe des zu erstellenden Dump-Volumes an. Der Wert auto bewirkt, dass die Standard-Swap-Größe verwendet wird. Sie können eine Größe angeben, indem Sie einen *size*-Wert verwenden. Sofern nicht g (GB) angegeben wird, gilt für die Größenangabe die Einheit MB.

vdev-Liste Gibt eines oder mehrere Speichergeräte an, aus denen der Pool gebildet wird. Das Format der *vdev-Liste* ist identisch mit dem Format für den Befehl zpool create. Derzeit werden bei der Angabe mehrerer Geräte nur gespiegelte Konfigurationen unterstützt. Bei den Geräten in *vdevlist* muss es sich um Bereiche für den Root-Pool handeln. Der Wert any bewirkt, dass die Installationssoftware ein passendes Gerät wählt.

Sie können beliebig viele Festplatten spiegeln. Die Größe des erstellten Pools richtet sich jedoch nach der kleinsten der angegebenen Festplatten. Weitere Informationen zum Erstellen von gespiegelten Speicher-Pools finden Sie unter „Speicher-Pools mit Datenspiegelung“ auf Seite 69.

JumpStart-Profilbeispiele für ZFS

Dieser Abschnitt enthält einige Beispiele für ZFS-spezifische JumpStart-Profile.

Das folgende Profil führt eine Erstinstallation (angegeben mit install_type *initial_install*) in einem neuen Pool (angegeben mit pool *newpool*) durch, dessen Größe gemäß dem Schlüsselwort auto automatisch auf die Größe der angegebenen Festplatten bemessen wird. Die Größe des Swap-Bereichs und des Dump-Geräts in einer Festplattenkonfiguration mit Datenspiegelung wird automatisch mit dem Schlüsselwort auto festgelegt (mit dem Schlüsselwort mirror und Angabe der Datenträger als *c0t0d0s0* und *c0t1d0s0*). Die Merkmale der Boot-Umgebung werden mit dem Schlüsselwort bootenv festgelegt, eine neue BU mit dem Schlüsselwort installbe installiert und der bename mit dem Wert *s10-xx* wird erstellt.

```
install_type initial_install
pool newpool auto auto auto mirror c0t0d0s0 c0t1d0s0
bootenv installbe bename s10-xx
```

Das folgende Profil führt eine Erstinstallation mit dem Schlüsselwort install_type *initial_install* des Metaclusters SUNWCall in einem neuen Pool namens newpool durch, dessen

Größe 80 GB beträgt. Dieser Pool wird mit einem Swap-Volume von 2 GB und einem Dump-Volume von ebenfalls 2 GB in einer gespiegelten Konfiguration zwei beliebiger verfügbarer Geräte erstellt, die groß genug sind, um ein Pool mit einer Kapazität von 80 GB zu erstellen. Sollten zwei derartige Geräte nicht verfügbar sein, schlägt die Installation fehl. Die Merkmale der Boot-Umgebung werden mit dem Schlüsselwort `bootenv` festgelegt, eine neue BU mit dem Schlüsselwort `installbe` installiert und der `bename` mit dem Wert *s10-xx* wird erstellt.

```
install_type initial_install
cluster SUNWCall
pool newpool 80g 2g 2g mirror any any
bootenv installbe bename s10-xx
```

Die JumpStart-Installationssyntax ermöglicht Ihnen, ein UFS-Dateisystem auf einem Datenträger, der auch einen ZFS-Root-Pool enthält, zu erhalten oder zu erstellen. Diese Konfiguration wird nicht für Produktionssysteme empfohlen, kann aber für die Erfordernisse der Umstellung kleinerer Systeme (beispielsweise eines Laptops) verwendet werden.

JumpStart-Probleme im Zusammenhang mit ZFS

Vor Beginn einer JumpStart-Installation eines boot-fähigen ZFS-Root-Dateisystems sind folgende Probleme zu berücksichtigen.

- Ein vorhandener ZFS-Speicher-Pool kann nicht für eine JumpStart-Installation zum Erstellen eines boot-fähigen ZFS-Root-Dateisystems verwendet werden. Sie müssen einen neuen ZFS-Speicher-Pool mit Syntax wie der folgenden erstellen:

  ```
  pool rpool 20G 4G 4G c0t0d0s0
  ```

- Sie müssen den Pool mit Festplattenbereichen anstatt mit gesamten Festplatten erstellen. Siehe hierzu „Oracle Solaris-Installation und Oracle Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung“ auf Seite 127. Die im folgenden Beispiel gezeigte fett gedruckte Syntax ist unzulässig:

  ```
  install_type initial_install
  cluster SUNWCall
  pool rpool all auto auto mirror c0t0d0 c0t1d0
  bootenv installbe bename newBE
  ```

  Die im folgenden Beispiel gezeigte fett gedruckte Syntax ist zulässig:

  ```
  install_type initial_install
  cluster SUNWCall
  pool rpool all auto auto mirror c0t0d0s0 c0t1d0s0
  bootenv installbe bename newBE
  ```

Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem (Oracle Solaris Live Upgrade)

Die Oracle Solaris Live Upgrade-Funktionen aus früheren Versionen sind weiterhin verfügbar und verhalten sich in Bezug auf UFS-Komponenten unverändert.

Folgende Funktionen stehen ebenfalls zur Verfügung:

- Für die Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem muss mit der Option `-p` ein vorhandener ZFS-Speicher-Pool bestimmt werden.
- Wenn das UFS-Root-Dateisystem Komponenten auf verschiedenen Bereichen umfasst, werden diese in den ZFS-Root-Pool überführt.
- Sie können ein System mit Zonen zwar migrieren, die unterstützten Konfigurationen sind in Solaris 10 10/08 jedoch beschränkt. Ab Solaris 10 5/09 werden mehr Zonenkonfigurationen unterstützt. Weitere Informationen finden Sie in den folgenden Abschnitten:
  - „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 150
  - „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (ab Solaris 10 5/09)“ auf Seite 156

  Informieren Sie sich unter „Migration in ein ZFS-Root-Dateisystem mit Oracle Solaris Live Upgrade (ohne Zonen) “ auf Seite 146, wenn Sie ein ZFS-Root-Dateisystem ohne Zonen migrieren möchten.
- Oracle Solaris Live Upgrade kann die ZFS-Snapshot- und -Klon-Funktionen verwenden, wenn Sie eine neue ZFS-BU in demselben Pool erstellen. Dadurch wird die BU-Erstellung wesentlich schneller als in vorherigen Solaris-Versionen.

Ausführliche Informationen zu den Oracle Solaris-Installations- und Oracle Solaris Live Upgrade-Funktionen finden Sie in *Oracle Solaris 10 9/10 Installationshandbuch: Solaris Live Upgrade und Planung von Upgrades* .

Die Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem läuft in folgenden Grundschritten ab:

- Installieren Sie Solaris 10 10/08, Solaris 10 5/09, Solaris 10 10/09 oder Oracle Solaris 10 9/10 oder verwenden Sie das Standard-Upgrade-Programm, um eine Aktualisierung einer frühren Solaris 10-Version auf einem beliebigen unterstützten SPARC- oder x 86-System durchzuführen.
- Wenn Sie Solaris 10 10/08 oder eine höhere Version ausführen, erstellen Sie einen ZFS-Speicher-Pool für das ZFS-Root-Dateisystem.
- Verwenden Sie Oracle Solaris Live Upgrade, um das UFS-Root-Dateisystem in ein ZFS-Root-Dateisystem zu migrieren.

- Aktivieren der ZFS-BU mit dem Befehl `luactivate`.

Informationen zu Voraussetzungen für ZFS und Oracle Solaris Live Upgrade finden Sie unter „Oracle Solaris-Installation und Oracle Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung“ auf Seite 127.

Probleme bei der ZFS-Migration mit Oracle Solaris Live Upgrade

Für die Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Oracle Solaris Live Upgrade sind folgende Probleme zu berücksichtigen:

- Die standardmäßige Upgrade-Option der grafischen Oracle Solaris-Installationsoberfläche ist für die Migration von einem UFS- in ein ZFS-Root-Dateisystem nicht verfügbar. Hierzu müssen Sie Oracle Solaris Live Upgrade verwenden.
- Vor dem Oracle Solaris Live Upgrade müssen Sie den ZFS-Speicher-Pool erstellen, der zum Booten verwendet werden soll. Aufgrund von derzeitigen Boot-Einschränkungen muss der ZFS-Root-Pool außerdem mit Bereichen anstatt mit ganzen Festplatten erstellt werden. Beispiel:

  ```
  # zpool create rpool mirror c1t0d0s0 c1t1d0s0
  ```

  Vergewissern Sie sich vor dem Erstellen des neuen Pools, dass die für den Pool vorgesehenen Festplatten ein SMI-Label (VTOC) anstatt eines EFI-Labels aufweisen. Bei einer Neubezeichnung der Festplatte mit einem SMI-Label ist darauf zu achten, dass durch die Bezeichnung keine Änderungen am Partitionierungsschema erfolgt sind. In den meisten Fällen sollte die gesamte Festplattenkapazität in den für den Root-Pool vorgesehenen Bereichen liegen.
- Die Erstellung einer UFS-BU aus einer ZFS-BU ist mit Oracle Solaris Live Upgrade nicht möglich. Wenn Sie eine UFS-BU in eine ZFS-BU migrieren und die UFS-BU beibehalten, kann sowohl aus der UFS-BU als auch der ZFS-BU gebootet werden.
- Benennen Sie die ZFS-BUs nicht mit dem Befehl `zfs rename` um, da Oracle Solaris Live Upgrade diese Namensänderung nicht erkennt. Nachfolgende Befehle wie z. B. `ludelete` schlagen fehl. Sehen Sie davon ab, ZFS-Pools oder -Dateisysteme umzubenennen, wenn Sie bereits vorhandene BUs weiterhin verwenden möchten.
- Wenn Sie eine alternative BU erstellen, die ein Klon der primären BU ist, können Sie nicht auf die Optionen `-f`, `-x`, `-y`, `-Y` und `-z` zum Einbinden oder Ausschließen von Dateien in die bzw. aus der primären BU zurückgreifen. Diese Optionen zum Einbinden und Ausschließen können weiterhin in folgenden Fällen verwendet werden:

  ```
  UFS -> UFS
  UFS -> ZFS
  ZFS -> ZFS (different pool)
  ```

- Während die Aktualisierung eines UFS-Root-Dateisystems auf ein ZFS-Root-Dateisystem mit Oracle Solaris Live Upgrade möglich ist, können Sie mit Oracle Solaris Live Upgrade keine Aktualisierung von Nicht-Root- oder freigegebenen Dateisystemen vornehmen.
- Der Befehl `lu` kann nicht zum Erstellen bzw. Migrieren eines ZFS-Root-Dateisystems verwendet werden.

Migration in ein ZFS-Root-Dateisystem mit Oracle Solaris Live Upgrade (ohne Zonen)

Die folgenden Beispiele veranschaulichen die Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem.

Wenn Sie ein System mit Zonen migrieren oder aktualisieren, lesen Sie folgende Abschnitte:

- „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)" auf Seite 150
- „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (ab Solaris 10 5/09)" auf Seite 156

BEISPIEL 5–3 Migrieren eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Oracle Solaris Live Upgrade

Das folgende Beispiel zeigt, wie eine BU eines ZFS-Root-Dateisystems aus einem UFS-Root-Dateisystem erstellt wird. Die aktuelle BU `ufsBE`, die ein UFS-Root-Dateisystem enthält, wird mit der Option `-c` angegeben. Wenn Sie die Option `- c` nicht angeben, wird als BU-Name standardmäßig der Gerätename verwendet. Die neue BU `zfsBE` wird mit der Option `-n` angegeben. Vor der Ausführung des `lucreate`-Vorgangs muss bereits ein ZFS-Speicher-Pool vorhanden sein.

Damit ein ZFS-Speicher-Pool upgrade- und bootfähig ist, muss er auf Datenträger-Speicherbereichen und darf nicht auf einer gesamten Festplatte erstellt werden. Vergewissern Sie sich vor dem Erstellen des neuen Pools, dass die für den Pool vorgesehenen Festplatten ein SMI-Label (VTOC) anstatt eines EFI-Labels aufweisen. Bei einer Neubezeichnung der Festplatte mit einem SMI-Label ist darauf zu achten, dass durch die Bezeichnung keine Änderungen am Partitionierungsschema erfolgt sind. In den meisten Fällen sollte die gesamte Festplattenkapazität in den für den Root-Pool vorgesehenen Bereichen liegen.

```
# zpool create rpool mirror c1t2d0s0 c2t1d0s0
# lucreate -c ufsBE -n zfsBE -p rpool
Analyzing system configuration.
No name for current boot environment.
Current boot environment is named <ufsBE>.
Creating initial configuration for primary boot environment <ufsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
```

BEISPIEL 5–3 Migrieren eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Oracle Solaris Live Upgrade *(Fortsetzung)*

```
PBE configuration successful: PBE name <ufsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <ufsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
The device </dev/dsk/c1t2d0s0> is not a root device for any boot environment; cannot get BE ID.
Creating configuration for boot environment <zfsBE>.
Source boot environment is <ufsBE>.
Creating boot environment <zfsBE>.
Creating file systems on boot environment <zfsBE>.
Creating <zfs> file system for </> in zone <global> on <rpool/ROOT/zfsBE>.
Populating file systems on boot environment <zfsBE>.
Checking selection integrity.
Integrity check OK.
Populating contents of mount point </>.
Copying.
Creating shared file system mount points.
Creating compare databases for boot environment <zfsBE>.
Creating compare database for file system </rpool/ROOT>.
Creating compare database for file system </>.
Updating compare databases on boot environment <zfsBE>.
Making boot environment <zfsBE> bootable.
Creating boot_archive for /.alt.tmp.b-qD.mnt
updating /.alt.tmp.b-qD.mnt/platform/sun4u/boot_archive
Population of boot environment <zfsBE> successful.
Creation of boot environment <zfsBE> successful.
```

Prüfen Sie nach Abschluss der `lucreate`-Vorgangs den BU-Status mithilfe des Befehls `lustatus`. Beispiel:

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufsBE                      yes      yes    yes       no     -
zfsBE                      yes      no     no        yes    -
```

Kontrollieren Sie dann die Liste der ZFS-Komponenten. Beispiel:

```
# zfs list
NAME                    USED  AVAIL  REFER  MOUNTPOINT
rpool                  7.17G  59.8G  95.5K  /rpool
rpool/ROOT             4.66G  59.8G    21K  /rpool/ROOT
rpool/ROOT/zfsBE       4.66G  59.8G  4.66G  /
rpool/dump                2G  61.8G    16K  -
rpool/swap              517M  60.3G    16K  -
```

Aktivieren Sie dann mit dem Befehl `luactivate` die neue ZFS-BU. Beispiel:

```
# luactivate zfsBE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfsBE>.
```

BEISPIEL 5–3 Migrieren eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Oracle Solaris Live Upgrade *(Fortsetzung)*

```
**********************************************************************

The target boot environment has been activated. It will be used when you
reboot. NOTE: You MUST NOT USE the reboot, halt, or uadmin commands. You
MUST USE either the init or the shutdown command when you reboot. If you
do not use either init or shutdown, the system will not boot using the
target BE.

**********************************************************************
.
.
.
Modifying boot archive service
Activation of boot environment <zfsBE> successful.
```

Als Nächstes starten Sie das System in der ZFS-BU neu.

```
# init 6
```

Vergewissern Sie sich, dass die ZFS-BU aktiv ist:

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufsBE                      yes      no     no        yes    -
zfsBE                      yes      yes    yes       no     -
```

Wenn Sie wieder auf die UFS-BU zurückschalten, müssen Sie während des Bootens der ZFS-BU erstellte ZFS-Speicher-Pools zurückimportieren, da sie nicht automatisch in der UFS-BU verfügbar sind.

Wird die UFS-BU nicht mehr benötigt, kann Sie mit dem Befehl `ludelete` gelöscht werden.

BEISPIEL 5–4 Erstellen einer ZFS-BU aus einer ZFS-BU mit Oracle Solaris Live Upgrade

Das Erstellen einer ZFS-BU aus einer ZFS-BU in demselben Pool ist eine sehr schnelle Angelegenheit, da für diesen Vorgang die ZFS-Snapshot- und -Klon-Funktionen verwendet werden. Wenn die aktuelle BU in demselben ZFS-Pool enthalten ist, wird die Option `-p` nicht berücksichtigt.

Wenn mehrere ZFS-BUs vorhanden sind, gehen Sie wie folgt vor, um auszuwählen, aus welcher BU gebootet werden soll:

- SPARC: Sie können mit dem Befehl `boot -L` die verfügbaren BUs ermitteln und mit dem Befehl `boot -Z` die BU auswählen, aus der gebootet werden soll.
- x86: Sie können eine BU aus dem GRUB-Menü auswählen.

BEISPIEL 5–4 Erstellen einer ZFS-BU aus einer ZFS-BU mit Oracle Solaris Live Upgrade *(Fortsetzung)*

Weitere Informationen finden Sie unter Beispiel 5–9.

```
# lucreate -n zfs2BE
Analyzing system configuration.
No name for current boot environment.
INFORMATION: The current boot environment is not named - assigning name <zfsBE>.
Current boot environment is named <zfsBE>.
Creating initial configuration for primary boot environment <zfsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <zfsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <zfsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs2BE>.
Source boot environment is <zfsBE>.
Creating boot environment <zfs2BE>.
Cloning file systems from boot environment <zfsBE> to create boot environment <zfs2BE>.
Creating snapshot for <rpool/ROOT/zfsBE> on <rpool/ROOT/zfsBE@zfs2BE>.
Creating clone for <rpool/ROOT/zfsBE@zfs2BE> on <rpool/ROOT/zfs2BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs2BE>.
Population of boot environment <zfs2BE> successful.
Creation of boot environment <zfs2BE> successful.
```

BEISPIEL 5–5 Upgrade der ZFS-BU (`luupgrade`)

Sie können die ZFS-BU mithilfe von zusätzliche Packages oder Patches aktualisieren.

Dieser Vorgang läuft im Wesentlichen wie folgt ab:

- Erstellen Sie mit dem Befehl `lucreate` eine alternative BU.
- Aktivieren Sie die alternative BU und booten Sie.
- Fügen Sie der primären ZFS-BU mit dem Befehl `luupgrade` die Packages oder Patches hinzu.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      no     no        yes    -
zfs2BE                     yes      yes    yes       no     -
# luupgrade -p -n zfsBE -s /net/system/export/s10up/Solaris_10/Product SUNWchxge
Validating the contents of the media </net/install/export/s10up/Solaris_10/Product>.
Mounting the BE <zfsBE>.
Adding packages to the BE <zfsBE>.

Processing package instance <SUNWchxge> from </net/install/export/s10up/Solaris_10/Product>

Chelsio N110 10GE NIC Driver(sparc) 11.10.0,REV=2006.02.15.20.41
Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
```

BEISPIEL 5–5 Upgrade der ZFS-BU (`luupgrade`) *(Fortsetzung)*

```
This appears to be an attempt to install the same architecture and
version of a package which is already installed.  This installation
will attempt to overwrite this package.

Using </a> as the package base directory.
## Processing package information.
## Processing system information.
   4 package pathnames are already properly installed.
## Verifying package dependencies.
## Verifying disk space requirements.
## Checking for conflicts with packages already installed.
## Checking for setuid/setgid programs.

This package contains scripts which will be executed with super-user
permission during the process of installing this package.

Do you want to continue with the installation of <SUNWchxge> [y,n,?] y
Installing Chelsio N110 10GE NIC Driver as <SUNWchxge>

## Installing part 1 of 1.
## Executing postinstall script.

Installation of <SUNWchxge> was successful.
Unmounting the BE <zfsBE>.
The package add to the BE <zfsBE> completed.
```

Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)

Sie können ein System mit Zonen mithilfe von Oracle Solaris Live Upgrade zwar migrieren, die unterstützten Konfigurationen sind in Solaris 10 10/08 jedoch beschränkt. Nach der Installation von Solaris 10 5/09 oder einer höheren Version oder der Aktualisierung auf Solaris 10 5/09 oder eine höhere Version werden mehr Zonenkonfigurationen unterstützt. Weitere Informationen finden Sie unter „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (ab Solaris 10 5/09)“ auf Seite 156.

In diesem Abschnitt wird beschrieben, wie Sie ein System mit Zonen installieren und konfigurieren müssen, damit es mit Solaris Oracle Live Upgrade aktualisiert und gepatcht werden kann. Informieren Sie sich unter „Migration in ein ZFS-Root-Dateisystem mit Oracle Solaris Live Upgrade (ohne Zonen) “ auf Seite 146, wenn Sie ein ZFS-Root-Dateisystem ohne Zonen migrieren möchten.

Wenn Sie ein System mit Zonen migrieren möchten bzw. die Konfiguration eines Systems mit Zonen in Betracht ziehen, berücksichtigen·Sie in Solaris 10 10/08 die folgenden Vorgehensweisen:

- „So migrieren Sie ein UFS-Root-Dateisystem mit Zonen-Roots auf UFS auf ein ZFS-Root-Dateisystem (Solaris 10 10/08)" auf Seite 151
- „So konfigurieren Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS (Solaris 10 10/08)" auf Seite 153
- „So aktualisieren bzw. patchen Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS (Solaris 10 10/08)" auf Seite 154
- „Lösen von Problemen mit ZFS-Einhängepunkten, die ein erfolgreiches Booten verhindern (Solaris 10 10/08)" auf Seite 174

Halten Sie sich an die empfohlenen Vorgehensweisen zum Einrichten von Zonen mit einem ZFS-Root-Dateisystem, damit Sie für dieses System Oracle Solaris Live Upgrade verwenden können.

▼ So migrieren Sie ein UFS-Root-Dateisystem mit Zonen-Roots auf UFS auf ein ZFS-Root-Dateisystem (Solaris 10 10/08)

Dieses Verfahren erklärt, wie ein UFS-Root-Dateisystem mit installierten Zonen in ein ZFS-Root-Dateisystem migriert wird, damit die ZFS-Zonen-Root-Konfiguration aktualisiert oder gepatcht werden kann.

In den nachfolgenden Schritten ist der Name des Beispiel-Pools `rpool`, und der Name der aktiven Boot-Umgebung ist `s10BE*`.

1 **Rüsten·Sie das System auf Solaris 10 10/08 auf, falls darauf ein früheres Solaris 10-Release installiert ist.**

Weitere Informationen zum Aktualisieren eines Systems, auf dem Solaris 10 installiert ist, finden Sie in *Oracle Solaris 10 9/10 Installationshandbuch: Solaris Live Upgrade und Planung von Upgrades* .

2 **Erstellen Sie den Root-Pool.**

```
# zpool create rpool mirror c0t1d0 c1t1d0
```

Informationen zu Voraussetzungen für den Root-Pool finden Sie unter „Oracle Solaris-Installation und Oracle Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung" auf Seite 127.

3 **Überprüfen Sie, ob die Zonen aus der UFS-Umgebung gebootet wurden.**

4 **Erstellen Sie die neue ZFS-Boot-Umgebung.**

```
# lucreate -n s10BE2 -p rpool
```

Dieser Befehl erstellt im Root-Pool Datasets für die neue Boot-Umgebung und kopiert die aktuelle Boot-Umgebung einschließlich der Zonen in diese Datasets.

5 **Aktivieren Sie die neue ZFS-Boot-Umgebung.**

```
# luactivate s10BE2
```

Jetzt besitzt das System ein ZFS-Root-Dateisystem, die Zonen-Roots auf UFS befinden sich jedoch noch im UFS-Root-Dateisystem. Als nächster Schritt müssen die UFS-Zonen vollständig in eine unterstützte ZFS-Konfiguration migriert werden.

6 **Starten Sie das System neu.**

```
# init 6
```

7 **Migrieren Sie die Zonen in eine ZFS-BU.**

a. **Booten Sie die Zonen.**

b. **Erstellen Sie eine weitere ZFS-BU innerhalb des Pools.**

```
# lucreate s10BE3
```

c. **Aktivieren Sie die neue Boot-Umgebung.**

```
# luactivate s10BE3
```

d. **Starten Sie das System neu.**

```
# init 6
```

Dieser Schritt stellt sicher, dass die ZFS-BU und die Zonen gebootet werden können.

8 **Beseitigen Sie alle potenziellen Probleme mit Einhängepunkten.**

Aufgrund eines Fehlers im Oracle Solaris Live Upgrade kann das Booten der nichtaktiven Boot-Umgebung fehlschlagen, wenn ein ZFS-Dataset oder ein ZFS-Dataset einer Zone in der Boot-Umgebung einen ungültigen Einhängepunkt besitzt.

a. **Überprüfen Sie die Ausgabe des Befehls** `zfs list`**.**

Suchen Sie ungültige temporäre Einhängepunkte. Beispiel:

```
# zfs list -r -o name,mountpoint rpool/ROOT/s10u6

NAME                               MOUNTPOINT
rpool/ROOT/s10u6                   /.alt.tmp.b-VP.mnt/
rpool/ROOT/s10u6/zones             /.alt.tmp.b-VP.mnt//zones
rpool/ROOT/s10u6/zones/zonerootA   /.alt.tmp.b-VP.mnt/zones/zonerootA
```

Der Einhängepunkt für die ZFS-Root-BU (`rpool/ROOT/s10u6`) muss `/` sein.

b. Setzen Sie die Einhängepunkte für die ZFS-BU und ihre Datasets zurück.

Beispiel:

```
# zfs inherit -r mountpoint rpool/ROOT/s10u6
# zfs set mountpoint=/ rpool/ROOT/s10u6
```

c. Starten Sie das System neu.

Wählen Sie die Boot-Umgebung aus, deren Einhängepunkte Sie gerade korrigiert haben, wenn im GRUB-Menü bzw. in der Eingabeaufforderung des OpenBoot-PROM die Option zum Booten einer spezifischen Boot-Umgebung angezeigt wird.

▼ So konfigurieren Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS (Solaris 10 10/08)

In diesem Verfahren wird erklärt, wie ein ZFS-Root-Dateisystem und eine ZFS-Zonen-Root-Konfiguration einrichtet werden, die aktualisiert oder gepatcht werden können. In dieser Konfiguration werden die ZFS-Zonen-Roots als ZFS-Datasets erstellt.

In den nachfolgenden Schritten ist der Name des Beispiel-Pools `rpool`, und der Name der aktiven Boot-Umgebung ist `s10BE`. Der Name für das Zonen-Dataset kann ein beliebiger zulässiger Dataset-Name sein. Der Name des Zonen-Dataset im folgenden Beispiel ist `zones`.

1 **Installieren Sie mithilfe der interaktiven Solaris-Textmodus-Installationsoption oder des Solaris JumpStart-Installationsverfahrens ein System mit ZFS-Root.**

Weitere Informationen zur Installation eines ZFS-Root-Dateisystems mithilfe des Erstinstallationsverfahrens bzw. mit Solaris JumpStart finden Sie unter „Installieren eines ZFS-Root-Dateisystems (Erstinstallation)“ auf Seite 130 oder „Installieren eines ZFS-Root-Dateisystems (Oracle Solaris JumpStart-Installation)“ auf Seite 140.

2 **Booten Sie das System vom neu erstellten Root-Pool.**

3 **Erstellen Sie ein Dataset zum Gruppieren der Zonen-Roots.**

Beispiel:

```
# zfs create -o canmount=noauto rpool/ROOT/s10BE/zones
```

Das Setzen des Wertes `noauto` der Eigenschaft `canmount` garantiert, dass das Dataset nur von der speziellen Aktion von Oracle Solaris Live Upgrade und dem Startup-Code des Systems eingehängt werden kann.

4 **Hängen Sie das neu erstellte Zonen-Dataset ein.**

```
# zfs mount rpool/ROOT/s10BE/zones
```

Das Dataset wird unter `/zones` eingehängt.

5 **Erstellen Sie für jede Zonen-Root ein Dataset und hängen Sie dieses ein.**

```
# zfs create -o canmount=noauto rpool/ROOT/s10BE/zones/zonerootA
# zfs mount rpool/ROOT/s10BE/zones/zonerootA
```

6 **Setzen Sie für das Zonen-Root-Verzeichnis die entsprechenden Zugriffsrechte.**

```
# chmod 700 /zones/zonerootA
```

7 **Konfigurieren Sie die Zone und setzen Sie den Zonen-Pfad wie folgt:**

```
# zonecfg -z zoneA
    zoneA: No such zone configured
    Use 'create' to begin configuring a new zone.
    zonecfg:zoneA> create
    zonecfg:zoneA> set zonepath=/zones/zonerootA
```

Mithilfe der folgenden Syntax können Sie festlegen, dass die Zonen beim Booten des Systems automatisch gebootet werden sollen:

```
zonecfg:zoneA> set autoboot=true
```

8 **Installieren Sie die Zone.**

```
# zoneadm -z zoneA install
```

9 **Booten Sie die Zone.**

```
# zoneadm -z zoneA boot
```

▼ So aktualisieren bzw. patchen Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS (Solaris 10 10/08)

Verwenden Sie dieses Verfahren, wenn Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS aktualisieren bzw. patchen müssen. Bei solchen Aktualisierungen kann es sich um eine Systemaufrüstung oder das Anwenden von Patches handeln.

In den nachfolgenden Schritten ist `newBE` der Beispielname der Boot-Umgebung, die aktualisiert bzw. gepatcht werden soll.

1 **Erstellen Sie die Boot-Umgebung, die aktualisiert bzw. gepatcht werden soll.**

```
# lucreate -n newBE
```

Die vorhandene Boot-Umgebung sowie sämtliche Zonen werden geklont. Für jedes Dataset der ursprünglichen Boot-Umgebung wird ein Dataset erstellt. Die neuen Datasets werden im gleichen Pool erstellt, in dem sich auch der aktuelle Root-Pool befindet.

2 **Wählen Sie eines der folgenden Verfahren aus, um das System aufzurüsten oder Patches auf die neue Boot-Umgebung anzuwenden:**

- Rüsten Sie das System auf.

```
# luupgrade -u -n newBE -s /net/install/export/s10u7/latest
```

Die Option -s gibt an, wo sich der Solaris-Installationsdatenträger befindet.

- Wenden Sie auf die neue Boot-Umgebung Patches an.

```
# luupgrade -t -n newBE -t -s /patchdir 139147-02 157347-14
```

3 **Aktivieren Sie die neue Boot-Umgebung.**

```
# luactivate newBE
```

4 **Booten Sie das System von der neu aktivierten Boot-Umgebung.**

```
# init 6
```

5 **Beseitigen Sie alle potenziellen Probleme mit Einhängepunkten.**

Aufgrund eines Fehlers im Oracle Solaris Live Upgrade kann das Booten der nichtaktiven Boot-Umgebung fehlschlagen, wenn ein ZFS-Dataset oder ein ZFS-Dataset einer Zone in der Boot-Umgebung einen ungültigen Einhängepunkt besitzt.

a. **Überprüfen Sie die Ausgabe des Befehls `zfs list`.**

Suchen Sie ungültige temporäre Einhängepunkte. Beispiel:

```
# zfs list -r -o name,mountpoint rpool/ROOT/newBE

NAME                               MOUNTPOINT
rpool/ROOT/newBE                   /.alt.tmp.b-VP.mnt/
rpool/ROOT/newBE/zones             /.alt.tmp.b-VP.mnt/zones
rpool/ROOT/newBE/zones/zonerootA   /.alt.tmp.b-VP.mnt/zones/zonerootA
```

Der Einhängepunkt für die ZFS-Root-BU (`rpool/ROOT/newBE`) muss `/` sein.

b. **Setzen Sie die Einhängepunkte für die ZFS-BU und ihre Datasets zurück.**

Beispiel:

```
# zfs inherit -r mountpoint rpool/ROOT/newBE
# zfs set mountpoint=/ rpool/ROOT/newBE
```

c. **Starten Sie das System neu.**

Wählen Sie die Boot-Umgebung aus, deren Einhängepunkte Sie gerade korrigiert haben, wenn im GRUB-Menü bzw. in der Eingabeaufforderung des OpenBoot-PROM die Option zum Booten einer spezifischen Boot-Umgebung angezeigt wird.

Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (ab Solaris 10 5/09)

Ab Solaris 10 10/08 können Sie Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen verwenden. Zusätzliche Konfigurationen von Sparse-Root- und Gesamt-Root-Zonen werden durch Live Upgrade ab Solaris 10 5/09 unterstützt.

In diesem Abschnitt wird beschrieben, wie Sie ein System mit Zonen konfigurieren, damit es mit Oracle Solaris Live Upgrade ab Solaris 10 5/09 aktualisiert und gepatcht werden kann. Informieren Sie sich unter „Migration in ein ZFS-Root-Dateisystem mit Oracle Solaris Live Upgrade (ohne Zonen) " auf Seite 146, wenn Sie ein ZFS-Root-Dateisystem ohne Zonen migrieren möchten.

Berücksichtigen Sie folgende Punkte, wenn Sie Oracle Solaris Live Upgrade für ZFS und Zonen ab Solaris 10 5/09 verwenden:

- Wenn Sie Oracle Solaris Live Upgrade für in Solaris 10 5/09 unterstützte Zonenkonfigurationen verwenden, müssen Sie zunächst mit dem standardmäßigen Upgrade-Programm eine Aktualisierung auf Solaris 10 5/09 durchführen.
- Dann können Sie mit Oracle Solaris Live Upgrade Ihr UFS-Root-Dateisystem mit Zonen-Roots in ein ZFS-Root-Dateisystem migrieren oder ein Upgrade oder Patch auf Ihr ZFS-Root-Dateisystem und Zonen-Roots anwenden.
- Nicht unterstützte Zonenkonfigurationen aus einer früheren Solaris 10-Version können nicht direkt in Solaris 10 5/09 migriert werden.

Anhand der folgende Vorgehensweisen können Sie mit Solaris ab Version 10 5/09 ein System mit Zonen migrieren oder konfigurieren:

- „Unterstütztes ZFS mit Zonen-Root-Konfigurationsinformationen (ab Solaris 10 5/09)" auf Seite 156
- „So erstellen Sie eine ZFS-BU mit einem ZFS-Root-Dateisystem und einem Zonen-Root (ab Solaris 10 5/09)" auf Seite 158
- „So aktualisieren oder patchen Sie ein ZFS-Root-Dateisystem mit Zonen-Roots (ab Solaris 10 5/09)" auf Seite 160
- „So migrieren Sie ein UFS-Root-Dateisystem mit Zonen-Roots in ein ZFS-Root-Dateisystem (ab Solaris 10 5/09)" auf Seite 163

Unterstütztes ZFS mit Zonen-Root-Konfigurationsinformationen (ab Solaris 10 5/09)

Überprüfen Sie die unterstützten Zonenkonfigurationen, bevor Sie Oracle Solaris Live Upgrade zur Migration oder zum Upgrade eines Systems mit Zonen verwenden.

- **Migrieren eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem** – Die folgenden Zonen-Root-Konfigurationen werden unterstützt:
  - In einem Verzeichnis des UFS-Root-Dateisystems
  - In einem Unterverzeichnis eines Einhängepunkts im UFS-Root-Dateisystem
  - UFS-Root-Dateisystem mit einem Zonen-Root in einem UFS-Root-Dateisystem-Verzeichnis eines UFS-Root-Dateisystem-Einhängepunkts und ein ZFS-Nicht-Root-Pool mit einem Zonen-Root

  Die folgende UFS-/Zonen-Konfiguration wird nicht unterstützt: UFS-Root-Dateisystem mit einem Zonen-Root als Einhängepunkt.
- **Migrieren oder Aktualisieren eines UFS-Root-Dateisystems auf ein ZFS-Root-Dateisystem** – Die folgenden Zonen-Root-Konfigurationen werden ununterstützt:
  - In einem Dataset im ZFS-Root-Pool. In einigen Fällen erstellt Oracle Solaris Live Upgrade ein Dataset für das Zonen-Root (`zoneds`), wenn vor dem Oracle Solaris Live Upgrade kein solches Dataset vorhanden ist.
  - In einem Unterverzeichnis des ZFS-Root-Dateisystems
  - In einem Dataset außerhalb des ZFS-Root-Dateisystems
  - In einem Unterverzeichnis eines Dataset außerhalb des ZFS-Root-Dateisystems
  - In einem Dataset in einem Pool außerhalb der Root. Im folgenden Beispiel ist `zonepool/zones` ein Dataset, das die Zonen-Roots enthält, und `rpool` enthält die ZFS-BU:

    ```
    zonepool
    zonepool/zones
    zonepool/zones/myzone
    rpool
    rpool/ROOT
    rpool/ROOT/myBE
    ```

    Mit der folgenden Syntax erstellt Oracle Solaris Live Upgrade Snapshots und Klone in `zonepool` und der BU `rpool`:

    ```
    # lucreate -n newBE
    ```

    Die BU `newBE` in `rpool/ROOT/newBE` wird erstellt. Wenn die BU `newBE` aktiviert ist, bietet sie Zugriff auf die `zonepool`-Komponenten.

    Wäre `/zonepool/zones` im vorherigen Beispiel ein Unterverzeichnis und kein separates Dataset, dann würde Live Upgrade es als Komponente des Root-Pools `rpool` migrieren.
- **Informationen zur Migration oder Aktualisierung von UFS- und ZFS-Zonen** – Berücksichtigen Sie folgende Informationen zur Migration oder Aktualisierung einer UFS- oder ZFS-Umgebung:

- Wenn Sie Ihre Zonen wie unter „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 150 in Solaris 10 10/08 konfiguriert und eine Aktualisierung auf Solaris 10 5/09 oder eine höhere Version durchgeführt haben, sollten Sie in ein ZFS-Root-Dateisystem migrieren oder Solaris Live Upgrade für eine Aktualisierung auf Solaris 10 5/09 oder eine höhere Version verwenden können.
- Erstellen Sie keine Zonen-Roots in verschachtelten Verzeichnissen wie `zones/zone1` und `zones/zone1/zone2`. Anderenfalls kann das Einhängen während des Boot-Vorgangs fehlschlagen.

▼ So erstellen Sie eine ZFS-BU mit einem ZFS-Root-Dateisystem und einem Zonen-Root (ab Solaris 10 5/09)

Verwenden Sie dieses Verfahren, nachdem Sie eine Erstinstallation von Solaris 10 5/09 oder einer höheren Version durchgeführt haben, um ein ZFS-Root-Dateisystem zu erstellen. Verwenden Sie dieses Verfahren ebenfalls, nachdem Sie die Funktion `luupgrade` verwendet haben, um ein ZFS-Root-Dateisystem auf Solaris 10 5/09 oder eine höhere Version zu aktualisieren. Eine mit dieser Vorgehensweise erstellte ZFS-BU kann aktualisiert oder gepatcht werden.

Das in den folgenden Schritten verwendete Oracle Solaris 10 9/10-Beispielsystem besitzt ein ZFS-Root-Dateisystem und ein Zonen-Root-Dataset in `/rpool/zones`. Eine ZFS-BU mit dem Namen `zfs2BE` wird erstellt, die aktualisiert oder gepatcht werden kann.

1 Überprüfen Sie die vorhandenen ZFS-Dateisysteme.

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
rpool                 7.26G  59.7G    98K  /rpool
rpool/ROOT            4.64G  59.7G    21K  legacy
rpool/ROOT/zfsBE      4.64G  59.7G  4.64G  /
rpool/dump            1.00G  59.7G  1.00G  -
rpool/export            44K  59.7G    23K  /export
rpool/export/home       21K  59.7G    21K  /export/home
rpool/swap               1G  60.7G    16K  -
rpool/zones            633M  59.7G   633M  /rpool/zones
```

2 Stellen Sie sicher, dass die Zonen installiert und gebootet sind.

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /rpool/zones                   native   shared
```

3 Erstellen Sie die ZFS-BU.

```
# lucreate -n zfs2BE
Analyzing system configuration.
No name for current boot environment.
INFORMATION: The current boot environment is not named - assigning name <zfsBE>.
```

```
Current boot environment is named <zfsBE>.
Creating initial configuration for primary boot environment <zfsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <zfsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <zfsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs2BE>.
Source boot environment is <zfsBE>.
Creating boot environment <zfs2BE>.
Cloning file systems from boot environment <zfsBE> to create boot environment <zfs2BE>.
Creating snapshot for <rpool/ROOT/zfsBE> on <rpool/ROOT/zfsBE@zfs2BE>.
Creating clone for <rpool/ROOT/zfsBE@zfs2BE> on <rpool/ROOT/zfs2BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs2BE>.
Population of boot environment <zfs2BE> successful.
Creation of boot environment <zfs2BE> successful.
```

4 **Aktivieren Sie die ZFS-BU.**

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      yes    yes       no     -
zfs2BE                     yes      no     no        yes    -
# luactivate zfs2BE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfs2BE>.
.
.
.
# init 6
```

5 **Überprüfen Sie, ob die ZFS-Dateisysteme und Zonen in der neuen BU erstellt wurden.**

```
# zfs list
NAME                               USED  AVAIL  REFER  MOUNTPOINT
rpool                             7.38G  59.6G    98K  /rpool
rpool/ROOT                        4.72G  59.6G    21K  legacy
rpool/ROOT/zfs2BE                 4.72G  59.6G  4.64G  /
rpool/ROOT/zfs2BE@zfs2BE          74.0M      -  4.64G  -
rpool/ROOT/zfsBE                  5.45M  59.6G  4.64G  /.alt.zfsBE
rpool/dump                        1.00G  59.6G  1.00G  -
rpool/export                        44K  59.6G    23K  /export
rpool/export/home                   21K  59.6G    21K  /export/home
rpool/swap                           1G  60.6G    16K  -
rpool/zones                       17.2M  59.6G   633M  /rpool/zones
rpool/zones-zfsBE                  653M  59.6G   633M  /rpool/zones-zfsBE
rpool/zones-zfsBE@zfs2BE          19.9M      -   633M  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - zfszone          installed  /rpool/zones                   native   shared
```

So aktualisieren oder patchen Sie ein ZFS-Root-Dateisystem mit Zonen-Roots (ab Solaris 10 5/09)

Verwenden Sie dieses Verfahren, wenn Sie ein ZFS-Root-Dateisystem mit Zonen-Roots in Solaris 10 5/09 oder einer höheren Version aktualisieren oder patchen müssen. Bei solchen Aktualisierungen·kann es sich um eine Systemaufrüstung oder das Anwenden von Patches handeln.

In den nachfolgenden Schritten ist `zfs2BE` der Beispielname der Boot-Umgebung, die aktualisiert·bzw. gepatcht werden soll.

1 Überprüfen Sie die vorhandenen ZFS-Dateisysteme.

```
# zfs list
NAME                               USED  AVAIL  REFER  MOUNTPOINT
rpool                             7.38G  59.6G   100K  /rpool
rpool/ROOT                        4.72G  59.6G    21K  legacy
rpool/ROOT/zfs2BE                 4.72G  59.6G  4.64G  /
rpool/ROOT/zfs2BE@zfs2BE          75.0M      -  4.64G  -
rpool/ROOT/zfsBE                  5.46M  59.6G  4.64G  /
rpool/dump                        1.00G  59.6G  1.00G  -
rpool/export                        44K  59.6G    23K  /export
rpool/export/home                   21K  59.6G    21K  /export/home
rpool/swap                          1G   60.6G    16K  -
rpool/zones                       22.9M  59.6G   637M  /rpool/zones
rpool/zones-zfsBE                  653M  59.6G   633M  /rpool/zones-zfsBE
rpool/zones-zfsBE@zfs2BE          20.0M      -   633M  -
```

2 Stellen Sie sicher, dass die Zonen installiert und gebootet sind.

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   5 zfszone          running    /rpool/zones                   native   shared
```

3 Erstellen Sie die ZFS-BU, die aktualisiert oder gepatcht werden soll.

```
# lucreate -n zfs2BE
Analyzing system configuration.
Comparing source boot environment <zfsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs2BE>.
Source boot environment is <zfsBE>.
Creating boot environment <zfs2BE>.
Cloning file systems from boot environment <zfsBE> to create boot environment <zfs2BE>.
Creating snapshot for <rpool/ROOT/zfsBE> on <rpool/ROOT/zfsBE@zfs2BE>.
Creating clone for <rpool/ROOT/zfsBE@zfs2BE> on <rpool/ROOT/zfs2BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs2BE>.
Creating snapshot for <rpool/zones> on <rpool/zones@zfs10092BE>.
Creating clone for <rpool/zones@zfs2BE> on <rpool/zones-zfs2BE>.
Population of boot environment <zfs2BE> successful.
Creation of boot environment <zfs2BE> successful.
```

4 **Wählen Sie eines der folgenden Verfahren aus, um das System aufzurüsten oder Patches auf die neue Boot-Umgebung anzuwenden:**

- Rüsten·Sie das System auf.

  ```
  # luupgrade -u -n zfs2BE -s /net/install/export/s10up/latest
  ```

  Die Option -s gibt an, wo sich der Solaris-Installationsdatenträger befindet.

  Dieser Vorgang kann sehr viel Zeit beanspruchen.

  Ein vollständiges Beispiel für den `luupgrade`-Vorgang finden Sie in Beispiel 5–6.
- Wenden Sie auf die neue Boot-Umgebung Patches an.

  ```
  # luupgrade -t -n zfs2BE -t -s /patchdir patch-id-02 patch-id-04
  ```

5 **Aktivieren Sie die neue Boot-Umgebung.**

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      yes    yes       no     -
zfs2BE                     yes      no     no        yes    -
# luactivate zfs2BE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfs2BE>.
.
.
.
```

6 **Booten Sie das System von der neu aktivierten Boot-Umgebung.**

```
# init 6
```

Beispiel 5–6 Aktualisieren eines ZFS-Root-Dateisystems mit Zonen-Root auf ein ZFS-Root-Dateisystem in Oracle Solaris 9 10/10

In diesem Beispiel wird eine auf einem Solaris 9 10/10-System erstellte ZFS-BU (`zfsBE`) mit einem ZFS-Root-Dateisystem und einer Zonen-Root in einem Pool außerhalb des Root auf Oracle Solaris 10 10/09 aktualisiert. Dieser Vorgang kann viel Zeit beanspruchen. Dann wird die aktualisierte BU (`zfs2BE`) aktiviert. Stellen Sie vor dem Aktualisierungsversuch sicher, dass die Zonen installiert und gebootet wurden.

In diesem Beispiel werden der Pool `zonepool`, das Dataset `/zonepool/zones` und die Zone `zfszone` wie folgt erstellt:

```
# zpool create zonepool mirror c2t1d0 c2t5d0
# zfs create zonepool/zones
# chmod 700 zonepool/zones
# zonecfg -z zfszone
zfszone: No such zone configured
Use 'create' to begin configuring a new zone.
```

```
zonecfg:zfszone> create
zonecfg:zfszone> set zonepath=/zonepool/zones
zonecfg:zfszone> verify
zonecfg:zfszone> exit
# zoneadm -z zfszone install
cannot create ZFS dataset zonepool/zones: dataset already exists
Preparing to install zone <zfszone>.
Creating list of files to copy from the global zone.
Copying <8960> files to the zone.
.
.
.

# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /zonepool/zones                native   shared

# lucreate -n zfsBE
.
.
.
# luupgrade -u -n zfsBE -s /net/install/export/s10up/latest
40410 blocks
miniroot filesystem is <lofs>
Mounting miniroot at </net/system/export/s10up/latest/Solaris_10/Tools/Boot>
Validating the contents of the media </net/system/export/s10up/latest>.
The media is a standard Solaris media.
The media contains an operating system upgrade image.
The media contains <Solaris> version <10>.
Constructing upgrade profile to use.
Locating the operating system upgrade program.
Checking for existence of previously scheduled Live Upgrade requests.
Creating upgrade profile for BE <zfsBE>.
Determining packages to install or upgrade for BE <zfsBE>.
Performing the operating system upgrade of the BE <zfsBE>.
CAUTION: Interrupting this process may leave the boot environment unstable
or unbootable.
Upgrading Solaris: 100% completed
Installation of the packages from this media is complete.
Updating package information on boot environment <zfsBE>.
Package information successfully updated on boot environment <zfsBE>.
Adding operating system patches to the BE <zfsBE>.
The operating system patch installation is complete.
INFORMATION: The file </var/sadm/system/logs/upgrade_log> on boot
environment <zfsBE> contains a log of the upgrade operation.
INFORMATION: The file </var/sadm/system/data/upgrade_cleanup> on boot
environment <zfsBE> contains a log of cleanup operations required.
INFORMATION: Review the files listed above. Remember that all of the files
are located on boot environment <zfsBE>. Before you activate boot
environment <zfsBE>, determine if any additional system maintenance is
required or if additional media of the software distribution must be
installed.
The Solaris upgrade of the boot environment <zfsBE> is complete.
Installing failsafe
Failsafe install is complete.
# luactivate zfsBE
# init 6
```

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      no     no        yes    -
zfs2BE                     yes      yes    yes       no     -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - zfszone          installed  /zonepool/zones                native   shared
```

So migrieren Sie ein UFS-Root-Dateisystem mit Zonen-Roots in ein ZFS-Root-Dateisystem (ab Solaris 10 5/09)

Verwenden Sie dieses Verfahren, um ein System mit einem UFS-Root-Dateisystem und einer Zonen-Root auf Solaris 10 5/09 oder eine höhere Version zu migrieren. Erstellen Sie dann mit Oracle Solaris Live Upgrade eine ZFS-BU.

In den folgenden Schritten lautet der Name der UFS-Beispiel-BU `c0t1d0s0`, der UFS-Zonen-Root heißt `zonepool/zfszone` und die ZFS-Root-BU heißt `zfsBE`.

1 **Aktualisieren Sie das System auf Solaris 10 5/09 oder eine höhere Version, falls darauf eine frühere Solaris 10-Version installiert ist.**

Weitere Informationen zum Aktualisieren eines Systems, auf dem Solaris 10 installiert ist, finden Sie in *Oracle Solaris 10 9/10 Installationshandbuch: Solaris Live Upgrade und Planung von Upgrades* .

2 **Erstellen Sie den Root-Pool.**

Informationen zu Voraussetzungen für den Root-Pool finden Sie unter „Oracle Solaris-Installation und Oracle Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung“ auf Seite 127.

3 **Überprüfen Sie, ob die Zonen aus der UFS-Umgebung gebootet wurden.**

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /zonepool/zones                native   shared
```

4 **Erstellen Sie die neue ZFS-Boot-Umgebung.**

```
# lucreate -c c1t1d0s0 -n zfsBE -p rpool
```

Dieser Befehl erstellt im Root-Pool Datasets für die neue Boot-Umgebung und kopiert die aktuelle Boot-Umgebung einschließlich der Zonen in diese Datasets.

5 **Aktivieren Sie die neue ZFS-Boot-Umgebung.**

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
```

```
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
c1t1d0s0                   yes      no     no        yes    -
zfsBE                      yes      yes    yes       no     -         #
luactivate zfsBE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfsBE>.
.
.
.
```

6 Starten Sie das System neu.

```
# init 6
```

7 Überprüfen Sie, ob die ZFS-Dateisysteme und Zonen in der neuen BU erstellt wurden.

```
# zfs list
NAME                                USED  AVAIL  REFER  MOUNTPOINT
rpool                              6.17G  60.8G    98K  /rpool
rpool/ROOT                         4.67G  60.8G    21K  /rpool/ROOT
rpool/ROOT/zfsBE                   4.67G  60.8G  4.67G  /
rpool/dump                         1.00G  60.8G  1.00G  -
rpool/swap                          517M  61.3G    16K  -
zonepool                            634M  7.62G    24K  /zonepool
zonepool/zones                      270K  7.62G   633M  /zonepool/zones
zonepool/zones-c1t1d0s0             634M  7.62G   633M  /zonepool/zones-c1t1d0s0
zonepool/zones-c1t1d0s0@zfsBE       262K      -   633M  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - zfszone          installed  /zonepool/zones                native   shared
```

Beispiel 5–7 Migrieren eines UFS-Root-Dateisystems mit Zonen-Root in ein ZFS-Root-Dateisystem

In diesem Beispiel wird ein Oracle Solaris 9 10/10-System mit einem UFS-Root-Dateisystem, einem Zonen-Root (`/uzone/ufszone`), einem ZFS-Pool außerhalb des Roots (`pool`) und einem Zonen-Root (`/pool/zfszone`) in ein ZFS-Root-Dateisystem migriert. Stellen Sie vor dem Migrationsversuch sicher, dass der ZFS-Root-Pool erstellt sowie die Zonen installiert und gebootet wurden.

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 ufszone          running    /uzone/ufszone                 native   shared
   3 zfszone          running    /pool/zones/zfszone            native   shared
```

```
# lucreate -c ufsBE -n zfsBE -p rpool
Analyzing system configuration.
No name for current boot environment.
Current boot environment is named <zfsBE>.
Creating initial configuration for primary boot environment <zfsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <ufsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
```

```
Comparing source boot environment <ufsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
The device </dev/dsk/c1t1d0s0> is not a root device for any boot environment; cannot get BE ID.
Creating configuration for boot environment <zfsBE>.
Source boot environment is <ufsBE>.
Creating boot environment <zfsBE>.
Creating file systems on boot environment <zfsBE>.
Creating <zfs> file system for </> in zone <global> on <rpool/ROOT/zfsBE>.
Populating file systems on boot environment <zfsBE>.
Checking selection integrity.
Integrity check OK.
Populating contents of mount point </>.
Copying.
Creating shared file system mount points.
Copying root of zone <ufszone> to </.alt.tmp.b-EYd.mnt/uzone/ufszone>.
Creating snapshot for <pool/zones/zfszone> on <pool/zones/zfszone@zfsBE>.
Creating clone for <pool/zones/zfszone@zfsBE> on <pool/zones/zfszone-zfsBE>.
Creating compare databases for boot environment <zfsBE>.
Creating compare database for file system </rpool/ROOT>.
Creating compare database for file system </>.
Updating compare databases on boot environment <zfsBE>.
Making boot environment <zfsBE> bootable.
Creating boot_archive for /.alt.tmp.b-DLd.mnt
updating /.alt.tmp.b-DLd.mnt/platform/sun4u/boot_archive
Population of boot environment <zfsBE> successful.
Creation of boot environment <zfsBE> successful.
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufsBE                      yes      yes    yes       no     -
zfsBE                      yes      no     no        yes    -
# luactivate zfsBE
.
.
.
# init 6
.
.
.
# zfs list
NAME                                    USED  AVAIL  REFER  MOUNTPOINT
pool                                    628M  66.3G    19K  /pool
pool/zones                              628M  66.3G    20K  /pool/zones
pool/zones/zfszone                     75.5K  66.3G   627M  /pool/zones/zfszone
pool/zones/zfszone-ufsBE                628M  66.3G   627M  /pool/zones/zfszone-ufsBE
pool/zones/zfszone-ufsBE@zfsBE           98K      -   627M  -
rpool                                  7.76G  59.2G    95K  /rpool
rpool/ROOT                             5.25G  59.2G    18K  /rpool/ROOT
rpool/ROOT/zfsBE                       5.25G  59.2G  5.25G  /
rpool/dump                             2.00G  59.2G  2.00G  -
rpool/swap                              517M  59.7G    16K  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
```

```
  - ufszone          installed  /uzone/ufszone                native   shared
  - zfszone          installed  /pool/zones/zfszone           native   shared
```

ZFS-Unterstützung für Swap- und Dump-Geräte

Während einer Erstinstallation des Solaris-Betriebssystems oder eines Oracle Solaris Live Upgrades in einem UFS-Dateisystem wird auf einem ZFS-Volume im ZFS-Root-Pool ein Swap-Bereich erstellt. Beispiel:

```
# swap -l
swapfile                  dev  swaplo  blocks   free
/dev/zvol/dsk/rpool/swap 256,1      16 4194288 4194288
```

Während einer Erstinstallation des Solaris-Betriebssystems oder einem Oracle Solaris Live Upgrade in einem UFS-Dateisystem wird auf einem ZFS-Volume im ZFS-Root-Pool ein Dump-Gerät erstellt. Im Allgemeinen ist keine Verwaltung des Dump-Geräts erforderlich, da es während der Installation automatisch eingerichtet wird. Beispiel:

```
# dumpadm
      Dump content: kernel pages
       Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash/t2000
  Savecore enabled: yes
   Save compressed: on
```

Wenn Sie das Dump-Gerät deaktivieren und entfernen, müssen Sie es nach der Neuerstellung mit dem Befehl `dumpadm` aktivieren. In den meisten Fällen werden Sie nur die Größe des Dump-Geräts anpassen müssen. Dazu verwenden Sie den Befehl `zfs`.

Informationen zu den von den Installationsprogrammen erstellten Swap- und Dump-Volume-Größen finden Sie unter „Oracle Solaris-Installation und Oracle Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung" auf Seite 127.

Die Swap- und Dump-Volume-Größe kann während und nach der Installation geändert werden. Weitere Informationen finden Sie unter „Anpassen der Größe von ZFS-Swap- und Dump-Geräten" auf Seite 167.

Berücksichtigen Sie bei der Arbeit mit ZFS-Swap- und Dump-Geräten die folgenden Probleme:

- Für Swap-Bereich und Dump-Gerät müssen separate ZFS-Volumes verwendet werden.
- Derzeit wird die Verwendung von Swap-Dateien in ZFS-Dateisystemen nicht unterstützt.
- Wenn Sie den Swap-Bereich oder das Dump-Gerät nach der Installation oder dem Upgrade des Systems ändern müssen, benutzen Sie hierzu die Befehle `swap` und `dumpadm` wie in vorherigen Solaris-Versionen. Weitere Informationen finden Sie in Kapitel 20, „Configuring Additional Swap Space (Tasks)" in *System Administration Guide: Devices and*

File Systems und Kapitel 17, „Managing System Crash Information (Tasks)“ in *System Administration Guide: Advanced Administration*.

Weitere Informationen finden Sie in den folgenden Abschnitten:

- „Anpassen der Größe von ZFS-Swap- und Dump-Geräten“ auf Seite 167
- „Behebung von Problemen mit ZFS-Dump-Geräten“ auf Seite 169

Anpassen der Größe von ZFS-Swap- und Dump-Geräten

Da bei einer ZFS-Root-Installation anders mit der Größenbestimmung für Swap- und Dump-Geräte verfahren wird, müssen Sie unter Umständen vor, nach oder während der Installation diese Größen ändern.

- Die Größe der Swap- und Dump-Volumes kann während einer Erstinstallation angepasst werden. Weitere Informationen finden Sie unter Beispiel 5–1.
- Sie können vor der Ausführung eines Oracle Solaris Live Upgrade Swap- und Dump-Volumes erstellen und deren Größe festlegen. Beispiel:
  1. Erstellen Sie den Speicher-Pool.

     ```
     # zpool create rpool mirror c0t0d0s0 c0t1d0s0
     ```
  2. Erstellen Sie das Dump-Gerät.

     ```
     # zfs create -V 2G rpool/dump
     ```
  3. Aktivieren Sie das Dump-Gerät.

     ```
     # dumpadm -d /dev/zvol/dsk/rpool/dump
           Dump content: kernel pages
            Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
     Savecore directory: /var/crash/t2000
       Savecore enabled: yes
        Save compressed: on
     ```
  4. Wählen Sie eine der folgenden Optionen, um den Swap-Bereich zu erstellen:
     - SPARC: Erstellen Sie den Swap-Bereich. Stellen Sie die Blockgröße auf 8 KB ein.

       ```
       # zfs create -V 2G -b 8k rpool/swap
       ```
     - x86: Erstellen Sie den Swap-Bereich. Stellen Sie die Blockgröße auf 4 KB ein.

       ```
       # zfs create -V 2G -b 4k rpool/swap
       ```
  5. Wenn ein Swap-Gerät hinzugefügt oder verändert wird, müssen Sie das Swap-Volume aktivieren.
  6. Fügen Sie einen Eintrag für das Swap-Volume in die Datei `/etc/vfstab` ein.

Oracle Solaris Live Upgrade bietet keine Möglichkeit, die Größe vorhandener Swap- und Dump-Volumes zu ändern.

- Sie können nach der Installation des Systems die Eigenschaft `volsize` des Dump-Geräts zurücksetzen. Beispiel:

```
# zfs set volsize=2G rpool/dump
# zfs get volsize rpool/dump
NAME        PROPERTY  VALUE       SOURCE
rpool/dump  volsize   2G          -
```

- Sie können die Größe des Swap-Volume ändern, bis zur Integration von CR 6765386 wird jedoch empfohlen, das Swap-Gerät zunächst zu entfernen. Erstellen Sie es dann neu. Beispiel:

```
# swap -d /dev/zvol/dsk/rpool/swap
# zfs volsize=2G rpool/swap
# swap -a /dev/zvol/dsk/rpool/swap
```

  Informationen zum Entfernen eines Swap-Geräts auf einem aktiven System finden Sie unter:

  http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide

- Mit folgender Profilsyntax lässt sich die Größe der Swap- und Dump-Volumes in einem JumpStart-Profil anpassen:

```
install_type initial_install
cluster SUNWCXall
pool rpool 16g 2g 2g c0t0d0s0
```

  In diesem Profil werden die Größe des Swap-Volume und des Dump-Volume mit zwei `2g`-Einträgen auf je 2 GB festgelegt.

- Wenn Sie mehr Swap-Speicherplatz auf einem bereits installierten System benötigen, fügen Sie einfach ein weiteres Swap-Volume hinzu. Beispiel:

```
# zfs create -V 2G rpool/swap2
```

  Aktivieren Sie dann das neue Swap-Volume. Beispiel:

```
# swap -a /dev/zvol/dsk/rpool/swap2
# swap -l
swapfile                 dev  swaplo   blocks   free
/dev/zvol/dsk/rpool/swap  256,1      16 1058800 1058800
/dev/zvol/dsk/rpool/swap2 256,3      16 4194288 4194288
```

  Fügen Sie als Letztes einen zweiten Eintrag für das Swap-Volume in die Datei `/etc/vfstab` ein.

Behebung von Problemen mit ZFS-Dump-Geräten

Überprüfen Sie Folgendes, wenn Sie Probleme mit der Erfassung eines Systemabsturz-Speicherabzugs oder dem Ändern der Größe des Dump-Geräts haben.

- Wurde bei einem Systemabsturz nicht automatisch ein Speicherabzug erstellt, können Sie den Befehl `savecore` verwenden, um den Speicherabzug zu speichern.
- Wenn Sie erstmals ein ZFS-Root-Dateisystem erstellen oder in ein ZFS-Root-Dateisystem migrieren, wird automatisch ein Dump-Volume erstellt. In den meisten Fällen werden Sie nur die Größe des Dump-Volumes anpassen müssen, wenn die Größe des Standard-Dump-Volumes zu klein ist. Beispielsweise wird die Größe des Dump-Volumes in einem System mit hoher Speicherkapazität wie folgt auf 40 GB erhöht:

  ```
  # zfs set volsize=40G rpool/dump
  ```

  Das Ändern der Größe eine großen Dump-Volumes kann sehr zeitaufwändig sein.

  Wenn Sie aus irgendeinem Grunde ein Dump-Gerät aktivieren müssen, nachdem Sie ein Dump-Gerät manuell erstellt haben, verwenden Sie eine Syntax wie die folgende:

  ```
  # dumpadm -d /dev/zvol/dsk/rpool/dump
        Dump content: kernel pages
         Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
  Savecore directory: /var/crash/t2000
    Savecore enabled: yes
  ```
- Ein System mit einer Speicherkapazität von 128 GB oder mehr benötigt ein größeres Dump-Gerät als jenes, das standardmäßig erstellt wurde. Wenn das Dump-Gerät zu klein ist, um bei einem Systemabsturz einen Speicherabzug zu erfassen, wird eine Meldung wie die folgende angezeigt:

  ```
  # dumpadm -d /dev/zvol/dsk/rpool/dump
  dumpadm: dump device /dev/zvol/dsk/rpool/dump is too small to hold a system dump
  dump size 36255432704 bytes, device size 34359738368 bytes
  ```

  Informationen zum Ändern der Größe von Swap- und Dump-Geräten finden Sie unter „Planning for Swap Space“ in *System Administration Guide: Devices and File Systems*.
- Sie können momentan kein Dump-Gerät zu einem Pool mit mehreren Geräte der obersten Ebene hinzufügen. Eine Meldung ähnlich der Folgenden wird angezeigt:

```
# dumpadm -d /dev/zvol/dsk/datapool/dump
dump is not supported on device '/dev/zvol/dsk/datapool/dump': 'datapool' has multiple top level vdevs
```

  Fügen Sie das Dump-Gerät zum Root-Pool hinzu, der nicht mehrere Geräte der obersten Ebene enthalten kann.

Booten aus einem ZFS-Root-Dateisystem

Sowohl bei SPARC- als auch bei x86-Systemen kommt die neue Art des Bootens mit Boot-Archiv zum Einsatz. Dabei handelt es sich um ein Dateisystemabbild, in dem sich die zum Booten benötigten Dateien befinden. Beim Booten aus einem ZFS-Root-Dateisystem werden die Pfadnamen des Archivs und der Kernel-Datei in dem für das Booten ausgewählten Root-Dateisystem aufgelöst.

Wenn das System für die Installation gebootet wird, wird während des gesamten Installationsvorgangs ein RAM-Datenträger für das Root-Dateisystem verwendet.

Das Booten aus einem ZFS-Dateisystem unterscheidet sich vom Booten aus einem UFS-Dateisystem dadurch, dass ein Gerätebezeichner bei ZFS kein einzelnes Root-Dateisystem sondern einen Speicher-Pool kennzeichnet. Ein Speicher-Pool kann mehrere *boot-fähige Datasets* oder ZFS-Root-Dateisysteme enthalten. Beim Booten aus ZFS müssen Sie ein Boot-Gerät und ein Root-Dateisystem innerhalb des Pools angeben, das durch das Boot-Gerät identifiziert ist.

Standardmäßig ist das zum Booten ausgewählte Dataset das mit der Eigenschaft `bootfs` des Pools bezeichnete Dataset. Diese Standardauswahl kann durch Angabe eines alternativen boot-fähigen Datasets im Befehl `boot -Z` außer Kraft gesetzt werden.

Booten von einer alternativen Festplatte in einem gespiegelten ZFS-Root-Pool

Bei der Installation des Systems haben Sie die Möglichkeit, einen ZFS-Root-Pool mit Datenspiegelung zu erstellen oder eine Festplatte einzubinden, um nach der Installation ein solches zu erstellen. Weitere Informationen finden Sie unter:

- „Installieren eines ZFS-Root-Dateisystems (Erstinstallation)" auf Seite 130
- „So erstellen Sie einen gespiegelten Root-Pool (nach der Installation)" auf Seite 136

Beachten Sie die folgenden bekannten Probleme im Zusammenhang mit gespiegelten ZFS-Root-Pools:

- CR 6668666 – Um das Booten auf den anderen Festplatten in der Datenspiegelungskonfiguration zu ermöglichen, müssen Sie die Boot-Informationen auf den zusätzlich eingebundenen Datenträgern mithilfe des Befehls `installboot` oder `installgrub` installieren. Dieser Schritt ist unnötig, wenn Sie zur Erstellung des gespiegelten ZFS-Root-Pools die Erstinstallationsmethode verwenden. Wenn beispielsweise der Datenträger `c0t1d0s0` als zweiter Datenträger zur Datenspiegelungskonfiguration hinzugefügt wurde, lautet der Befehl `installboot` bzw. `installgrub` wie folgt:
  - SPARC:

```
sparc# installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c0t1d0s0
```

- x86:

  ```
  x86# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c0t1d0s0
  ```

- Es kann von verschiedenen Speichergeräten in einem gespiegelten ZFS-Root-Pool gebootet werden. Je nach der Hardwarekonfiguration kann es erforderlich sein, eine Aktualisierung von PROM oder BIOS vorzunehmen, um ein anderes Boot-Gerät angeben zu können.

  So ist es in diesem Pool beispielsweise möglich, vom Datenträger (`c1t0d0s0` oder `c1t1d0s0`) zu booten.

  ```
  # zpool status
    pool: rpool
   state: ONLINE
   scrub: none requested
  config:

          NAME          STATE     READ WRITE CKSUM
          rpool         ONLINE       0     0     0
            mirror-0    ONLINE       0     0     0
              c1t0d0s0  ONLINE       0     0     0
              c1t1d0s0  ONLINE       0     0     0
  ```

- SPARC: Geben Sie an der Eingabeaufforderung `ok` die alternative Festplatte ein.

  ```
  ok boot /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0
  ```

  Rufen Sie nach dem Neustart des Systems eine Bestätigung des aktiven Boot-Geräts ab. Beispiel:

  ```
  SPARC# prtconf -vp | grep bootpath
          bootpath:  '/pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a'
  ```

- x86: Wählen Sie aus dem entsprechenden BIOS-Menü eine alternative Festplatte im gespiegelten ZFS-Root-Pool aus.

  Anschließend verwenden Sie eine Syntax wie die folgende, um zu bestätigen, dass Sie von der alternativen Festplatte gebootet haben.

  ```
  x86# prtconf -v|sed -n '/bootpath/,/value/p'
          name='bootpath' type=string items=1
              value='/pci@0,0/pci8086,25f8@4/pci108e,286@0/disk@0,0:a'
  ```

SPARC: Booten aus einem ZFS-Root-Dateisystem

Auf SPARC-Systemen mit mehreren ZFS-Boot-Umgebungen können Sie mithilfe des Befehls `luactivate` von jeder Boot-Umgebung booten.

Während der Installation des Solaris-Betriebssystems und des Oracle Solaris Live Upgrade wird dem ZFS-Root-Dateisystem automatisch die Eigenschaft `bootfs` zugewiesen.

Innerhalb eines Pools können mehrere boot-fähige Datasets vorhanden sein. Standardmäßig bezieht sich die Eigenschaft `bootfs` des Pools auf den Eintrag für das boot-fähige Dataset in der Datei /*Pool-Name*/`boot/menu.lst` . Der Eintrag `menu.lst` kann jedoch den Befehl `bootfs` enthalten, der ein alternatives Dataset im Pool angibt. So ist es möglich, dass die Datei `menu.lst` Einträge für mehrere Root-Dateisysteme im Pool enthält.

Wenn auf einem System ein ZFS-Root-Dateisystem installiert oder eine Migration in ein ZFS-Root-Dateisystem vorgenommen wird, wird die Datei `menu.lst` um einen Eintrag wie diesen erweitert:

```
title zfsBE
bootfs rpool/ROOT/zfsBE
title zfs2BE
bootfs rpool/ROOT/zfs2BE
```

Bei Erstellung einer neuen BU wird die Datei `menu.lst` automatisch aktualisiert.

Auf SPARC-Systemen stehen zwei neue Boot-Optionen zur Verfügung:

- Nach der Aktivierung der BU können Sie mit dem Befehl boot `-L` eine Liste der boot-fähigen Datasets innerhalb eines ZFS-Pools anzeigen lassen. Anschließend können Sie eines der boot-fähigen Datasets in der Liste auswählen. Es werden ausführliche Anweisungen zum Booten dieses Datasets angezeigt. Befolgen Sie diese Anweisungen zum Booten des ausgewählten Datasets.
- Zum Booten eines bestimmten ZFS-Datasets können Sie den Befehl boot `-Z` *dataset* verwenden.

BEISPIEL 5–8 SPARC: Booten aus einer bestimmten ZFS-Boot-Umgebung

Wenn in einem ZFS-Speicher-Pool auf dem Boot-Gerät des Systems mehrere ZFS-BUs vorhanden sind, können Sie mit dem Befehl `luactivate` eine Standard-BU angeben.

So stehen beispielsweise die folgenden ZFS-BUs wie in der Ausgabe des Befehls `lustatus` beschrieben zur Verfügung:

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      no     no        yes    -
zfs2BE                     yes      yes    yes       no     -
```

Sind auf einem SPARC-System mehrere ZFS-BUs vorhanden, können Sie den Befehl `boot -L` verwenden, um aus einer BU zu booten, die sich von der Standard-BU unterscheidet. Jedoch wird eine BU, die aus einer `boot -L`-Sitzung gebootet wird, nicht auf die Standard-BU zurückgesetzt, und die Eigenschaft `bootfs` wird nicht aktualisiert. Wenn Sie die BU, die aus einer `boot -L` Sitzung gebootet wurde, zur Standard-BU machen möchten, müssen Sie sie mit dem Befehl `luactivate` aktivieren.

BEISPIEL 5–8 SPARC: Booten aus einer bestimmten ZFS-Boot-Umgebung *(Fortsetzung)*

Beispiel:

```
ok boot -L
Rebooting with command: boot -L
Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0  File and args: -L

1 zfsBE
2 zfs2BE
Select environment to boot: [ 1 - 2 ]: 1
To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/zfsBE

Program terminated
ok boot -Z rpool/ROOT/zfsBE
```

BEISPIEL 5–9 SPARC: Booten eines ZFS-Dateisystems im Failsafe-Modus

Auf SPARC-Systemen kann aus dem in `/platform/‘uname -i‘/failsafe` befindlichen Failsafe-Archiv gebootet werden. Gehen Sie dazu wie folgt vor:

```
ok boot -F failsafe
```

Um ein Failsafe-Archiv aus einem bestimmten boot-fähigen ZFS-Dataset zu booten, verwenden Sie eine Syntax wie die folgende:

```
ok boot -Z rpool/ROOT/zfsBE -F failsafe
```

x86: Booten aus einem ZFS-Root-Dateisystem

Während der Installation des Solaris-Betriebssystems oder des Oracle Solaris Live Upgrade werden der Datei */pool-name* `/boot/grub/menu.lst` die folgenden Einträge hinzugefügt, damit ZFS automatisch gebootet wird:

```
title Solaris 10 9/10  X86
findroot (rootfs0,0,a)
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS
module /platform/i86pc/boot_archive
title Solaris failsafe
findroot (rootfs0,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

Enthält das von GRUB als Boot-Gerät identifizierte Speichergerät einen ZFS-Speicher-Pool, wird auf Grundlage der Datei `menu.lst` das GRUB-Menü erstellt.

Auf x86-Systemen mit mehreren ZFS-BUs kann die BU über das GRUB-Menü gewählt werden. Ist das diesem Menüeintrag entsprechende Root-Dateisystem ein ZFS-Dataset, wird die folgende Option hinzugefügt:

```
-B $ZFS-BOOTFS
```

BEISPIEL 5–10 x86: Booten eines ZFS-Dateisystems

Wenn ein System aus einem ZFS-Dateisystem bootet, wird das Root-Gerät durch den Boot-Parameter `-B $ZFS-BOOTFS` in der Zeile `kernel` oder `module` des GRUB-Menüeintrags angegeben. Ebenso wie alle anderen über die Option `-B` angegebenen Parameter wird dieser Wert von GRUB an den Kernel übergeben. Beispiel:

```
title Solaris 10 9/10  X86
findroot (rootfs0,0,a)
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS
module /platform/i86pc/boot_archive
title Solaris failsafe
findroot (rootfs0,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

BEISPIEL 5–11 x86: Booten eines ZFS-Dateisystems im Failsafe-Modus

Das Failsafe-Archiv für x86-Systeme lautet `/boot/x86.miniroot-safe` und kann durch Auswahl des Solaris-Failsafe-Eintrags im GRUB-Menü gebootet werden. Beispiel:

```
title Solaris failsafe
findroot (rootfs0,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

Lösen von Problemen mit ZFS-Einhängepunkten, die ein erfolgreiches Booten verhindern (Solaris 10 10/08)

Die beste Methode zum Ändern der aktiven Boot-Umgebung ist der Befehl `luactivate`. Wenn das Booten von der aktiven Boot-Umgebung wegen eines ungültigen Patches oder eines Konfigurationsfehlers fehlschlägt, können Sie nur von einer anderen Boot-Umgebung booten, wenn Sie diese zur Boot-Zeit auswählen. Bei x86-System können Sie aus dem GRUB-Menü eine alternative Boot-Umgebung auswählen oder diese (bei SPARC-Systemen) explizit vom PROM booten.

Aufgrund eines Fehlers im Oracle Solaris Live Upgrade von Solaris 10 10/08 kann das Booten der nichtaktiven Boot-Umgebung fehlschlagen, wenn ein ZFS-Dataset oder ein ZFS-Dataset einer Zone in der Boot-Umgebung einen ungültigen Einhängepunkt besitzt. Der gleiche Bug verhindert auch das Einhängen einer BU, wenn sie ein separates `/var`-Dataset besitzt.

Wenn ein Zonen-Dataset einen ungültigen Einhängepunkt besitzt, kann dieser mit den folgenden Schritten korrigiert werden.

▼ So lösen Sie Probleme mit ZFS-Einhängepunkten

1. **Booten Sie das System von einem Failsafe-Archiv.**

2. **Importieren Sie den Pool.**

   Beispiel:

   ```
   # zpool import rpool
   ```

3. **Suchen Sie ungültige temporäre Einhängepunkte.**

   Beispiel:

   ```
   # zfs list -r -o name,mountpoint rpool/ROOT/s10u6

       NAME                               MOUNTPOINT
       rpool/ROOT/s10u6                   /.alt.tmp.b-VP.mnt/
       rpool/ROOT/s10u6/zones             /.alt.tmp.b-VP.mnt//zones
       rpool/ROOT/s10u6/zones/zonerootA   /.alt.tmp.b-VP.mnt/zones/zonerootA
   ```

   Der Einhängepunkt für die Root-BU (rpool/ROOT/s10u6) muss / sein.

   Wenn der Boot-Vorgang wegen Einhängeproblemen von /var fehlschlägt, sollten Sie einen ähnlichen unzulässigen Einhängepunkt für das /var-Dataset suchen.

4. **Setzen Sie die Einhängepunkte für die ZFS-BU und ihre Datasets zurück.**

   Beispiel:

   ```
   # zfs inherit -r mountpoint rpool/ROOT/s10u6
   # zfs set mountpoint=/ rpool/ROOT/s10u6
   ```

5. **Starten Sie das System neu.**

   Wählen Sie die Boot-Umgebung aus, deren Einhängepunkte Sie gerade korrigiert haben, wenn im GRUB-Menü bzw. in der Eingabeaufforderung des OpenBoot-PROM die Option zum Booten einer spezifischen Boot-Umgebung angezeigt wird.

Booten zur Wiederherstellung in einer ZFS-Root-Umgebung

Gehen Sie wie folgt vor, um das System zu booten und eine Wiederherstellung durchzuführen, wenn ein Root-Passwort verloren gegangen oder ein ähnliches Problem aufgetreten ist.

Je nach Schweregrad des Fehlers müssen Sie im Failsafe-Modus oder von einem alternativen Datenträger booten. Um ein verloren gegangenes oder unbekanntes Root-Passwort wiederherzustellen, können Sie in der Regel im Failsafe-Modus booten.

- „So booten Sie im ZFS-Failsafe-Modus“ auf Seite 176
- „So booten Sie ZFS von einem alternativen Datenträger“ auf Seite 176

Informationen zur Wiederherstellung eines Root-Pools oder Root-Pool-Snapshots finden Sie unter „Wiederherstellen von ZFS-Root-Pools oder Root-Pool-Snapshots“ auf Seite 177.

So booten Sie im ZFS-Failsafe-Modus

1. **Booten Sie im Failsafe-Modus.**

   Auf einem SPARC-System:

   ```
   ok boot -F failsafe
   ```

   Auf einem x86-System wählen Sie an der GRUB-Eingabeaufforderung den Failsafe-Modus aus.

2. **Hängen Sie das ZFS-BU in `/a` ein, wenn Sie dazu aufgefordert werden:**

   ```
   .
   .
   .
   ROOT/zfsBE was found on rpool.
   Do you wish to have it mounted read-write on /a? [y,n,?] y
   mounting rpool on /a
   Starting shell.
   ```

3. **Wechseln Sie in das Verzeichnis `/a/etc`.**

   ```
   # cd /a/etc
   ```

4. **Setzen Sie nötigenfalls den TERM-Typ.**

   ```
   # TERM=vt100
   # export TERM
   ```

5. **Korrigieren Sie die Datei `passwd` bzw. `shadow`.**

   ```
   # vi shadow
   ```

6. **Starten Sie das System neu.**

   ```
   # init 6
   ```

So booten Sie ZFS von einem alternativen Datenträger

Wenn ein Problem das erfolgreiche Booten des Systems verhindert oder ein anderes schwerwiegendes Problem auftritt, müssen Sie von einem Netzwerkinstallationsserver oder von einer Solaris-Installations-CD booten, den Root-Pool importieren, das ZFS-BU einhängen und anschließend versuchen, das Problem zu lösen.

1 **Booten Sie von einer Installations-CD oder über das Netzwerk.**

- SPARC:

```
ok boot cdrom -s
ok boot net -s
```

  Wenn Sie die Option -s nicht verwenden, müssen Sie das Installationsprogramm beenden.

- x86: Wählen Sie die Option zum Booten über das Netzwerk oder die Option zum Booten von einer lokalen CD.

2 **Importieren Sie den Root-Pool, und geben Sie einen alternativen Einhängepunkt an. Beispiel:**

```
# zpool import -R /a rpool
```

3 **Hängen Sie die ZFS-BU ein. Beispiel:**

```
# zfs mount rpool/ROOT/zfsBE
```

4 **Greifen Sie über das Verzeichnis** /a **auf den ZFS-BU-Inhalt zu.**

```
# cd /a
```

5 **Starten Sie das System neu.**

```
# init 6
```

Wiederherstellen von ZFS-Root-Pools oder Root-Pool-Snapshots

In den folgenden Abschnitten werden diese Vorgehensweisen beschrieben:

- „So ersetzen Sie eine Festplatte im ZFS-Root-Pool“ auf Seite 177
- „So erstellen Sie Root-Pool-Snapshots“ auf Seite 180
- „So erstellen Sie einen ZFS-Root-Pool neu und stellen Root-Pool-Snapshots wieder her“ auf Seite 181
- „So erstellen Sie nach dem Booten im Failsafe-Modus ein Dateisystem im Zustand eines früheren Snapshots wieder her“ auf Seite 183

▼ So ersetzen Sie eine Festplatte im ZFS-Root-Pool

Das Ersetzen einer Festplatte im Root-Pool kann aus folgenden Gründen erforderlich sein:

- Der Root-Pool ist zu klein und Sie möchten eine kleine durch eine größere Festplatte ersetzen.

- Eine Root-Pool-Festplatte ist ausgefallen. Wenn Sie einen nicht redundanten Pool verwenden und nach einem Festplattenausfall das System nicht mehr booten können, müssen Sie von einem anderen Medium wie einer CD oder dem Netzwerk booten, bevor Sie die Root-Pool-Festplatte ersetzen können.

In einer gespiegelten Root-Pool-Konfiguration können Sie versuchen, eine Festplatte zu ersetzen, ohne von einem alternativen Medium zu booten. Sie können eine ausgefallene Festplatte ersetzen, indem Sie den Befehl `zpool replace` verwenden. Wenn Sie über eine zusätzliche Festplatte verfügen, können Sie auch den Befehl `zpool attach` verwenden. In diesem Abschnitt finden Sie ein Beispiel für das Verfahren zum Einbinden einer zusätzlichen Festplatte und zum Entfernen einer Root-Pool-Festplatte.

Bei mancher Hardware müssen Sie eine Festplatte zunächst deaktivieren und dekonfigurieren, bevor Sie versuchen können, eine ausgefallene Festplatte mithilfe des Vorgangs `zpool replace` zu ersetzen. Beispiel:

```
# zpool offline rpool c1t0d0s0
# cfgadm -c unconfigure c1::dsk/c1t0d0
<Physically remove failed disk c1t0d0>
<Physically insert replacement disk c1t0d0>
# cfgadm -c configure c1::dsk/c1t0d0
# zpool replace rpool c1t0d0s0
# zpool online rpool c1t0d0s0
# zpool status rpool
<Let disk resilver before installing the boot blocks>
SPARC# installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c1t0d0s0
x86# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c1t9d0s0
```

Bei mancher Hardware müssen Sie die Ersatzfestplatte, nachdem diese eingesetzt wurde, weder aktivieren noch neu konfigurieren.

Sie müssen die Pfadnamen des Boot-Geräts der aktuellen und der neuen Festplatte angeben, damit Sie das Booten von der Ersatzfestplatte testen und manuell von der vorhandenen Festplatte booten können, falls das Booten von der Ersatzfestplatte fehlschlägt. In dem Beispiel des folgenden Verfahrens lautet der Pfadname für die aktuelle Root-Pool-Festplatte `c1t10d0s0`:

```
/pci@8,700000/pci@3/scsi@5/sd@a,0
```

Der Pfadname für die Ersatzfestplatte zum Booten (`c1t9d0s0`) lautet:

```
/pci@8,700000/pci@3/scsi@5/sd@9,0
```

1. **Verbinden Sie die Ersatzfestplatte (bzw. die neue Festplatte) physisch.**

2. **Vergewissern Sie sich, dass die neue Festplatte ein SMI-Label und den Bereich 0 hat.**

   Informationen zum Umbenennen einer für den Root-Pool bestimmten Festplatte finden Sie auf:

http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide

3 **Verbinden Sie die neue Festplatte mit dem Root-Pool.**

Beispiel:

```
# zpool attach rpool c1t10d0s0 c1t9d0s0
```

4 **Überprüfen Sie den Status des Root-Pools.**

Beispiel:

```
# zpool status rpool
  pool: rpool
 state: ONLINE
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress, 25.47% done, 0h4m to go
config:

        NAME           STATE     READ WRITE CKSUM
        rpool          ONLINE       0     0     0
          mirror-0     ONLINE       0     0     0
            c1t10d0s0  ONLINE       0     0     0
            c1t9d0s0   ONLINE       0     0     0

errors: No known data errors
```

5 **Wenden Sie nach Abschluss des Resilvering die Boot-Blöcke auf die neue Festplatte an.**

Verwenden Sie eine Syntax wie die folgende:

- SPARC:

  ```
  # installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c1t9d0s0
  ```
- x86:

  ```
  # installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c1t9d0s0
  ```

6 **Überprüfen Sie, ob Sie von der neuen Festplatte booten können.**

Auf einem SPARC-System würden Sie eine Syntax wie die folgende verwenden:

```
ok boot /pci@8,700000/pci@3/scsi@5/sd@9,0
```

7 **Wenn das System von der neuen Festplatte bootet, entfernen Sie die alte Festplatte.**

Beispiel:

```
# zpool detach rpool c1t10d0s0
```

8 **Richten Sie das System zum automatischen Booten von der neuen Festplatte ein, indem Sie den Befehl `eeprom`, den Befehl `setenv` am SPARC-Boot-PROM verwenden, oder das PC-BIOS neu konfigurieren.**

So erstellen Sie Root-Pool-Snapshots

Sie können Root-Pool-Snapshots zur Wiederherstellung erstellen. Der beste Weg zum Erstellen von Root-Pool-Snapshots besteht darin, einen rekursiven Snapshot des Root-Pools zu erstellen.

Im folgenden Verfahren wird ein rekursiver Root-Pool-Snapshot erstellt und in einer Datei innerhalb eines Pools auf einem entfernten System gespeichert. Wenn ein Root-Pool ausfällt, kann das entfernte Dataset durch Verwendung von NFS eingehängt und die Snapshot-Datei im wiederhergestellten Pool gespeichert werden. Sie können Root-Pool-Snapshots stattdessen als die eigentlichen Snapshots in einem Pool auf einem entfernten System speichern. Das Senden und Empfangen von Snapshots über ein entferntes System ist etwas komplexer, da Sie `ssh` konfigurieren oder `rsh` verwenden müssen, während das zu reparierende System von der Miniroot des Solaris-Betriebssystems gebootet wird.

Informationen zum Speichern und Wiederherstellen von Root-Pool-Snapshots auf entfernten Systemen und die aktuellsten Informationen zur Wiederherstellung von Root-Pools finden Sie unter:

`http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide`

Das Validieren von Snapshots, die auf entfernten Systemen als Dateien oder Snapshots gespeichert sind, ist ein wichtiger Schritt bei der Wiederherstellung von Root-Pools. Bei beiden Methoden sollten Snapshots regelmäßig wiederhergestellt werden, zum Beispiel, wenn sich die Konfiguration des Pools verändert oder das Solaris-Betriebssystem aktualisiert wird.

Im folgenden Verfahren wird das System von der Boot-Umgebung `zfsBE` gebootet.

1. **Erstellen Sie ein Pool- und Dateisystem zum Speichern von Snapshots auf einem entfernten System.**

   Beispiel:

   ```
   remote# zfs create rpool/snaps
   ```

2. **Geben Sie das Dateisystem für das lokale System frei.**

   Beispiel:

   ```
   remote# zfs set sharenfs='rw=local-system,root=local-system' rpool/snaps
   # share
   -@rpool/snaps   /rpool/snaps   sec=sys,rw=local-system,root=local-system   ""
   ```

3. **Erstellen Sie einen rekursiven Snapshot des Root-Pools.**

   ```
   local# zfs snapshot -r rpool@0804
   local# zfs list
   NAME                        USED  AVAIL  REFER  MOUNTPOINT
   rpool                      6.17G  60.8G    98K  /rpool
   rpool@0804                     0      -    98K  -
   rpool/ROOT                 4.67G  60.8G    21K  /rpool/ROOT
   rpool/ROOT@0804                0      -    21K  -
   ```

```
rpool/ROOT/zfsBE           4.67G  60.8G  4.67G  /
rpool/ROOT/zfsBE@0804       386K      -  4.67G  -
rpool/dump                 1.00G  60.8G  1.00G  -
rpool/dump@0804                0      -  1.00G  -
rpool/swap                  517M  61.3G    16K  -
rpool/swap@0804                0      -    16K  -
```

4 **Senden Sie die Root-Pool-Snapshots an das entfernte System.**

Beispiel:

```
local# zfs send -Rv rpool@0804 > /net/remote-system/rpool/snaps/rpool.0804
sending from @ to rpool@0804
sending from @ to rpool/swap@0804
sending from @ to rpool/ROOT@0804
sending from @ to rpool/ROOT/zfsBE@0804
sending from @ to rpool/dump@0804
```

▼ So erstellen Sie einen ZFS-Root-Pool neu und stellen Root-Pool-Snapshots wieder her

In diesem Verfahren wird von folgenden Voraussetzungen ausgegangen:

- Das ZFS-Root-Pool kann nicht wiederhergestellt werden.
- Die ZFS-Root-Pool-Snapshots wurden auf einem entfernten System gespeichert und über NFS freigegeben.

Alle Schritte werden auf dem lokalen System ausgeführt.

1 **Booten Sie das System über CD/DVD oder das Netzwerk.**

- SPARC: Wählen Sie eine der folgenden Boot-Methoden:

  ```
  ok boot net -s
  ok boot cdrom -s
  ```

  Wenn Sie die Option `-s` nicht verwenden, müssen Sie das Installationsprogramm beenden.
- x86: Wählen Sie die Option zum Booten über DVD oder das Netzwerk. Beenden Sie das Installationsprogramm.

2 **Hängen Sie das entfernte Snapshot-Dataset ein.**

Beispiel:

```
# mount -F nfs remote-system:/rpool/snaps /mnt
```

Wenn Ihre Netzwerkservices nicht konfiguriert sind, müssen Sie eventuell die IP-Adresse des *remote-system* angeben.

3 Wenn die Root-Pool-Festplatte ersetzt wird und keine Festplattenbezeichnung enthält, die von ZFS verwendet werden kann, müssen Sie die Festplatte umbenennen.

Weitere Informationen zum Umbenennen der Festplatte finden Sie auf der folgenden Site:

http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide

4 Erstellen Sie den Root-Pool neu.

Beispiel:

```
# zpool create -f -o failmode=continue -R /a -m legacy -o cachefile=
/etc/zfs/zpool.cache rpool c1t1d0s0
```

5 Stellen Sie die Root-Pool-Snapshots wieder her.

Dieser Schritt kann etwas Zeit beanspruchen. Beispiel:

```
# cat /mnt/rpool.0804 | zfs receive -Fdu rpool
```

Durch Verwenden der Option `-u` wird das wiederhergestellte Archiv nach dem `zfs receive`-Vorgang nicht eingehängt.

6 Überprüfen Sie, ob die Root-Pool-Datasets wiederhergestellt wurden.

Beispiel:

```
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
rpool                      6.17G  60.8G    98K  /a/rpool
rpool@0804                     0      -    98K  -
rpool/ROOT                 4.67G  60.8G    21K  /legacy
rpool/ROOT@0804                0      -    21K  -
rpool/ROOT/zfsBE           4.67G  60.8G  4.67G  /a
rpool/ROOT/zfsBE@0804       398K      -  4.67G  -
rpool/dump                 1.00G  60.8G  1.00G  -
rpool/dump@0804                0      -  1.00G  -
rpool/swap                  517M  61.3G    16K  -
rpool/swap@0804                0      -    16K  -
```

7 Legen Sie die Eigenschaft `bootfs` in der Root-Pool-BU fest.

Beispiel:

```
# zpool set bootfs=rpool/ROOT/zfsBE rpool
```

8 Installieren Sie auf der neuen Festplatte die Boot-Blöcke.

SPARC:

```
# installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c1t1d0s0
```

x86:

```
# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c1t1d0s0
```

9 Starten Sie das System neu.

```
# init 6
```

▼ So erstellen Sie nach dem Booten im Failsafe-Modus ein Dateisystem im Zustand eines früheren Snapshots wieder her

Für dieses Verfahren wird vorausgesetzt, dass Root-Pool-Snapshots verfügbar sind. In diesem Beispiel befinden sie sich auf dem lokalen System.

```
# zfs snapshot -r rpool@0804
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
rpool                      6.17G  60.8G    98K  /rpool
rpool@0804                     0      -    98K  -
rpool/ROOT                 4.67G  60.8G    21K  /rpool/ROOT
rpool/ROOT@0804                0      -    21K  -
rpool/ROOT/zfsBE           4.67G  60.8G  4.67G  /
rpool/ROOT/zfsBE@0804       398K      -  4.67G  -
rpool/dump                 1.00G  60.8G  1.00G  -
rpool/dump@0804                0      -  1.00G  -
rpool/swap                  517M  61.3G    16K  -
rpool/swap@0804                0      -    16K  -
```

1 **Fahren Sie das System herunter und booten Sie im Failsafe-Modus.**

```
ok boot -F failsafe
ROOT/zfsBE was found on rpool.
Do you wish to have it mounted read-write on /a? [y,n,?] y
mounting rpool on /a

Starting shell.
```

2 **Stellen Sie jeden Root-Pool-Snapshot wieder her.**

```
# zfs rollback rpool@0804
# zfs rollback rpool/ROOT@0804
# zfs rollback rpool/ROOT/zfsBE@0804
```

3 **Booten Sie erneut im Mehrbenutzer-Modus.**

```
# init 6
```



Verwalten von Oracle Solaris ZFS-Dateisystemen

Dieses Kapitel enthält ausführliche Informationen zum Verwalten von Oracle Solaris ZFS-Dateisystemen. Es werden Konzepte wie die hierarchische Dateisystemstrukturierung, Eigenschaftsvererbung, die automatische Verwaltung von Einhängepunkten sowie die Interaktion zwischen Netzwerkdateisystemen behandelt.

Dieses Kapitel enthält die folgenden Abschnitte:

Verwalten von ZFS-Dateisystemen (Übersicht)

ZFS-Dateisysteme setzen auf einem Speicher-Pool auf. Dateisysteme können dynamisch erstellt und gelöscht werden, ohne dass Festplattenspeicher zugewiesen oder formatiert werden muss. Da Dateisysteme so kompakt und der Dreh- und Angelpunkt der ZFS-Administration sind, werden Sie wahrscheinliche sehr viele davon erstellen.

ZFS-Dateisysteme werden mit dem Befehl `zfs` verwaltet. Der Befehl `zfs` enthält eine Reihe von Unterbefehlen, die spezifische Operationen an Dateisystemen ausführen. In diesem Kapitel werden diese Unterbefehle ausführlich beschrieben. Snapshots, Volumes und Klone werden von diesem Befehl ebenfalls verwaltet; die Leistungsmerkmale werden jedoch erst später behandelt. Ausführliche Informationen zu Snapshots und Klonen finden Sie in Kapitel 7, „Arbeiten mit Oracle Solaris ZFS-Snapshots und -Klonen". Ausführliche Informationen zu ZFS-Volumes finden Sie unter „ZFS-Volumes" auf Seite 287.

Hinweis – Der Begriff *Dataset* wird in diesem Kapitel als Oberbegriff für Dateisysteme, Snapshots, Klone und Volumes verwendet.

Erstellen, Entfernen und Umbenennen von ZFS-Dateisystemen

ZFS-Dateisysteme können mit den Befehlen `zfs create` bzw. `zfs destroy` erstellt und gelöscht werden. ZFS-Dateisysteme können mit dem Befehl `zfs rename` umbenannt werden.

- „Erstellen eines ZFS-Dateisystems“ auf Seite 186
- „Löschen eines ZFS-Dateisystems“ auf Seite 187
- „Umbenennen eines ZFS-Dateisystems“ auf Seite 188

Erstellen eines ZFS-Dateisystems

ZFS-Dateisysteme werden mit dem Befehl `zfs create` erstellt. Der Befehl `create` erfordert als einziges Argument den Namen des zu erstellenden Dateisystems. Der Name des Dateisystems wird wie folgt als Pfadname beginnend mit dem Namen des Pools angegeben:

Pool-Name/[Dateisystemname/]Dateisystemname

Der Pool-Name und die anfänglichen Dateisystemnamen im Pfad geben an, wo das neue Dateisystem in der Hierarchie erstellt wird. Der letzte Name im Pfad ist der Name des zu erstellenden Dateisystems. Der Dateisystemname muss den unter „Konventionen für das Benennen von ZFS-Komponenten“ auf Seite 51 aufgeführten Benennungskonventionen entsprechen.

Im folgenden Beispiel wird das Dateisystem `bonwick` im Dateisystem `tank/home` erstellt.

```
# zfs create tank/home/bonwick
```

ZFS hängt das neue Dateisystem bei fehlerfreier Erstellung automatisch ein. Dateisysteme werden standardmäßig als /*Dataset* eingehängt; dabei wird der im Unterbefehl `create` angegebene Pfad verwendet. In diesem Beispiel wird das neu erstellte Dateisystem `bonwick` unter `tank/home/bonwick` eingehängt. Weitere Informationen zu automatisch verwalteten Einhängepunkten finden Sie unter „Verwalten von ZFS-Einhängepunkten“ auf Seite 211.

Weitere Informationen zum Befehl `zfs create` finden Sie in der Man Page `zfs`(1M).

Sie können Dateisystemeigenschaften beim Erstellen des Dateisystems festlegen.

Im folgenden Beispiel wird ein Einhängepunkt von `/export/zfs` für das Dateisystem `tank/home` erstellt:

```
# zfs create -o mountpoint=/export/zfs tank/home
```

Weitere Informationen zu Eigenschaften von Dateisystemen finden Sie unter „ZFS-Eigenschaften“ auf Seite 189.

Löschen eines ZFS-Dateisystems

ZFS-Dateisysteme werden mit dem Befehl `zfs destroy` gelöscht. Das gelöschte Dateisystem wird automatisch für den Netzwerkzugriff gesperrt und ausgehängt. Weitere Informationen zur automatischen Verwaltung von Einhängepunkten und gemeinsam genutzten Objekten finden Sie unter „Automatische Einhängepunkte“ auf Seite 212.

Im folgenden Beispiel wird das Dateisystem `tabriz` gelöscht:

```
# zfs destroy tank/home/tabriz
```



Achtung – Beim Ausführen des Unterbefehls `destroy` wird keine Bestätigung des Löschvorgangs angefordert. Verwenden Sie diesen Befehl deshalb mit äußerster Vorsicht.

Wenn das zu löschende Dateisystem noch von Ressourcen verwendet wird und deswegen nicht ausgehängt werden kann, schlägt der Befehl `zfs destroy` fehl. Aktive Dateisysteme werden mit der Option `-f` gelöscht. Sie sollten diese Option mit Sorgfalt verwenden, da sie aktive Dateisysteme aushängt, für den Netzwerkzugriff sperrt und löscht und somit unvorgesehenes Systemverhalten verursachen kann.

```
# zfs destroy tank/home/ahrens
cannot unmount 'tank/home/ahrens': Device busy

# zfs destroy -f tank/home/ahrens
```

Der Befehl `zfs destroy` schlägt ebenfalls fehl, wenn in einem Dateisystem untergeordnete Dateisysteme vorhanden sind. Zum rekursiven Löschen von Dateisystemen und allen untergeordneten Dateisystemen dient die Option `-r`. Bitte beachten Sie, dass beim rekursiven Löschen auch Snapshots des Dateisystems gelöscht werden. Deshalb sollten Sie diese Option mit äußerster Vorsicht verwenden.

```
# zfs destroy tank/ws
cannot destroy 'tank/ws': filesystem has children
use '-r' to destroy the following datasets:
tank/ws/billm
tank/ws/bonwick
tank/ws/maybee

# zfs destroy -r tank/ws
```

Wenn das zu löschende Dateisystem indirekte untergeordnete Dateisysteme besitzt, schlägt auch der rekursive Löschbefehl fehl. Wenn Sie das Löschen *aller* untergeordneten Objekte

einschließlich geklonter Dateisysteme außerhalb der Zielhierarchie erzwingen wollen, müssen Sie die Option -R verwenden. Verwenden Sie diese Option mit äußerster Vorsicht.

```
# zfs destroy -r tank/home/schrock
cannot destroy 'tank/home/schrock': filesystem has dependent clones
use '-R' to destroy the following datasets:
tank/clones/schrock-clone

# zfs destroy -R tank/home/schrock
```



Achtung – Für die Optionen -f, - r und -R des Löschbefehls zfs destroy wird keine Bestätigung angefordert. Deshalb sollten Sie diese Optionen mit äußerster Vorsicht verwenden.

Weitere Informationen zu Snapshots und Klonen finden Sie in Kapitel 7, „Arbeiten mit Oracle Solaris ZFS-Snapshots und -Klonen“.

Umbenennen eines ZFS-Dateisystems

Dateisysteme können mit dem Befehl zfs rename umbenannt werden. Mit dem Unterbefehl rename können folgende Vorgänge ausgeführt werden:

- Ändern des Namens eines Dateisystems.
- Verlagern eines Dateisystems innerhalb der ZFS-Hierarchie
- Ändern des Namens eines Dateisystems und Verlagern dieses Systems innerhalb der ZFS-Hierarchie

Im folgende Beispiel wird der Unterbefehl rename verwendet, um ein Dateisystem von kustarz in kustarz_old umzubenennen:

```
# zfs rename tank/home/kustarz tank/home/kustarz_old
```

Das folgende Beispiel zeigt die Verwendung des Befehls zfs rename zum Verlagern eines ZFS-Dateisystems:

```
# zfs rename tank/home/maybee tank/ws/maybee
```

In diesem Beispiel wird das Dateisystem maybee von tank/home nach tank/ws verlagert. Wenn ein Dateisystem mithilfe des Umbenennungsbefehls verlagert wird, muss sich der neue Speicherort innerhalb des gleichen Pools befinden, und dieser muss über genügend Festplattenkapazität für das Dateisystem verfügen. Wenn der neue Speicherort nicht genügend Festplattenkapazität besitzt (z. B. weil das zugeteilte Kontingent erreicht ist), schlägt die Verlagerung mit rename fehl.

Weitere Informationen zu Kontingenten finden Sie unter „Einstellen von ZFS-Kontingenten und -Reservierungen“ auf Seite 218.

Durch die Umbenennung wird das betreffende Dateisystem mit allen seinen untergeordneten Dateisystemen ausgehängt und wieder neu eingehängt. Die Umbenennung schlägt fehl, wenn ein aktives Dateisystem nicht ausgehängt werden kann. Wenn dieses Problem auftritt, müssen Sie das Aushängen des Dateisystems erzwingen.

Informationen zum Umbenennen von Snapshots finden Sie unter „Umbenennen von ZFS-Snapshots" auf Seite 228.

ZFS-Eigenschaften

Mithilfe von Eigenschaften kann das Verhalten von Dateisystemen, Volumes, Snapshots und Klonen gesteuert werden Sofern nicht anders angegeben, gelten die in diesem Abschnitt beschriebenen Eigenschaften für alle Dataset-Typen.

- „Schreibgeschützte native ZFS-Eigenschaften" auf Seite 198
- „Konfigurierbare native ZFS-Eigenschaften" auf Seite 199
- „Benutzerdefinierte ZFS-Eigenschaften" auf Seite 202

Eigenschaften werden in zwei Typen (native und benutzerdefinierte Eigenschaften) eingeteilt. Native Eigenschaften exportieren interne Statistikinformationen und steuern das Systemverhalten von ZFS-Dateisystemen. Darüber hinaus können native Eigenschaften konfigurierbar oder schreibgeschützt sein. Benutzerdefinierte Eigenschaften wirken sich nicht auf das Verhalten von ZFS-Dateisystemen aus, können jedoch zum Versehen von Datasets mit Informationen, die für Ihre lokalen Gegebenheiten wichtig sind, verwendet werden. Weitere Informationen zu benutzerdefinierten Eigenschaften finden Sie unter „Benutzerdefinierte ZFS-Eigenschaften" auf Seite 202.

Die meisten konfigurierbaren Eigenschaften sind vererbbar. Vererbbare Eigenschaften werden von einem übergeordneten Dataset an alle seine untergeordneten Datasets weitergegeben.

Alle vererbbaren Eigenschaften haben eine Quelle, die angibt, wie eine Eigenschaft erhalten wurde. Die Eigenschaftsquelle kann die folgenden Werte besitzen:

`local`	Dieser Wert zeigt an, dass die Eigenschaft mithilfe des Befehls `zfs set` (siehe „Setzen von ZFS-Eigenschaften" auf Seite 206) explizit für das Dataset festgelegt wurde.
`inherited from` *Dataset-Name*	Dieser Wert zeigt an, dass die Eigenschaft von einem übergeordneten Objekt geerbt wurde.
`default`	Dieser Wert zeigt an, dass der Wert der betreffenden Eigenschaft weder geerbt noch lokal gesetzt wurde. Diese Quelle resultiert daraus, dass kein übergeordnetes Objekt eine Eigenschaft aufweist, die mit source `local` definiert ist.

In der folgenden Tabelle sind schreibgeschützte und konfigurierbare native Eigenschaften von ZFS-Dateisystemen aufgeführt. Schreibgeschützte Eigenschaften sind entsprechend gekennzeichnet. Alle anderen in dieser Tabelle aufgeführten nativen Eigenschaften sind konfigurierbar. Weitere Informationen zu benutzerdefinierten Eigenschaften finden Sie unter „Benutzerdefinierte ZFS-Eigenschaften" auf Seite 202.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften

Eigenschaft	Typ	Standardwert	Beschreibung
`aclinherit`	Zeichenkette	`secure`	Legt fest, wie Einträge in Zugriffssteuerungslisten beim Erstellen von Dateien und Verzeichnissen vererbt werden. Werte: `discard`, `noallow`, `secure` und `passthrough`. Eine Beschreibung dieser Werte finden Sie unter „Eigenschaften von Zugriffssteuerungslisten" auf Seite 249.
`aclmode`	Zeichenkette	`groupmask`	Legt fest, wie Einträge von Zugriffssteuerungslisten während eines `chmod`-Vorgangs geändert werden. Werte: `discard`, `groupmask` und `passthrough`. Eine Beschreibung dieser Werte finden Sie unter „Eigenschaften von Zugriffssteuerungslisten" auf Seite 249.
`atime`	Boolesch	`on`	Legt fest, ob beim Lesen von Dateien die Dateizugriffszeit aktualisiert wird. Durch Deaktivierung dieser Eigenschaft wird vermieden, dass während des Lesens von Dateien Datenverkehr entsteht, der aus Schreibvorgängen resultiert. Dadurch kann die Leistung erheblich verbessert werden. E-Mail-Programme und ähnliche Dienstprogramme können allerdings in ihrer Funktion beeinträchtigt werden.
`available`	Zahl	entf.	Diese schreibgeschützte Eigenschaft gibt die für ein Dataset und alle seine untergeordneten Objekte verfügbare Festplattenkapazität an. Dabei wird angenommen, dass im Pool keine Aktivität vorliegt. Da Festplattenkapazität innerhalb eines Pools gemeinsam genutzt wird, kann die verfügbare Kapazität durch verschiedene Faktoren wie z. B. physische Speicherkapazität des Pools, Kontingente, Reservierungen oder andere im Pool befindliche Datasets beschränkt werden. Die Abkürzung der Eigenschaft lautet `avail`. Weitere Informationen zur Berechnung von Festplattenkapazität finden Sie unter „Berechnung von ZFS-Festplattenkapazität" auf Seite 62.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`canmount`	Boolesch	`on`	Legt fest, ob ein Dateisystem mit dem Befehl `zfs mount` eingehängt werden kann. Diese Eigenschaft ist für jedes Dateisystem einstellbar und kann nicht vererbt werden. Durch Setzen dieser Eigenschaft auf `off` kann jedoch an untergeordnete Dateisysteme ein Einhängepunkt vererbt werden. Die Dateisysteme selbst werden jedoch nicht eingehängt. Wenn die Option `noauto` gesetzt ist, kann ein Dataset nur explizit ein- oder ausgehängt werden. Das Dataset wird weder beim Erstellen oder Importieren automatisch noch mit dem Befehl `zfs mount- a` eingehängt oder mit dem Befehl `zfs unmount-a` ausgehängt. Weitere Informationen finden Sie unter „Die Eigenschaft `canmount`" auf Seite 201.
`checksum`	Zeichenkette	`on`	Aktiviert/deaktiviert die Prüfsumme zur Validierung der Datenintegrität. Der Standardwert ist `on`. Dadurch wird automatisch ein geeigneter Algorithmus (gegenwärtig `fletcher4`) gesetzt. Werte: `on`, `off`, `fletcher2` , `fletcher4` und `sha256`. Der Wert `off` deaktiviert die Integritätsprüfung von Benutzerdaten. Der Wert `off` wird nicht empfohlen.
`compression`	Zeichenkette	`off`	Aktiviert oder deaktiviert die Komprimierung für ein Dataset. Werte: `on`, `off`, `lzjb`, `gzip` und `gzip-` *N*. Derzeit hat das Setzen der Eigenschaft auf `lzjb`, `gzip` oder `gzip-` *N* dieselbe Wirkung wie die Einstellung auf `on`. Durch Aktivieren der Komprimierung an einem Dateisystem mit bereits vorhandenen Daten werden nur neu hinzugekommene Daten komprimiert. Vorhandene Daten bleiben unkomprimiert. Die Abkürzung der Eigenschaft lautet `compress`
`compressratio`	Zahl	entf.	Schreibgeschützte Eigenschaft, die das für das betreffende Dataset erreichte Komprimierungsverhältnis als Faktor ausdrückt. Die Komprimierung kann durch Ausführen von `zfs set compression=on` *dataset* aktiviert werden. Der Wert wird aus der logischen Kapazität aller Dateien und der Kapazität der entsprechend referenzierten physischen Daten berechnet. Explizite Speicherplatzeinsparungen durch die Eigenschaft `compression` sind in diesem Wert enthalten.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`copies`	Zahl	`1`	Legt die Anzahl der Kopien von Benutzerdaten pro Dateisystem fest. Verfügbare Werte sind `1`, `2` oder `3`. Diese Kopien werden zusätzlich zu etwaigen Redundanzfunktionen auf Pool-Ebene angelegt. Die von zusätzlichen Kopien beanspruchte Festplattenkapazität wird auf die entsprechende Datei bzw. das Dataset angerechnet und zählt für Kontingente und Reservierungen. Darüber hinaus wird die Eigenschaft `used` entsprechend aktualisiert, wenn das Erstellen mehrerer Kopien aktiviert wurde. Sie sollten diese Eigenschaft beim Erstellen des Dateisystems setzen, da sich das Ändern dieser Eigenschaft bei einem bereits vorhandenen Dateisystem nur auf neu geschriebene Daten auswirkt.
`creation`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft, die angibt, wann ein Dataset erstellt wurde (Datum/Uhrzeit).
`devices`	Boolesch	`on`	Legt fest, ob Gerätedateien in einem Dateisystem geöffnet werden können.
`exec`	Boolesch	`on`	Legt fest, ob Programme innerhalb eines Dateisystems ausführbar sind. Wenn diese Eigenschaft auf `off` gesetzt ist, werden `mmap(2)`-Aufrufe mit `PROT_EXEC` nicht zugelassen.
`mounted`	Boolesch	entf.	Schreibgeschützte Eigenschaft, die angibt, ob gegenwärtig ein Dateisystem, Klon oder Snapshot eingehängt ist. Diese Eigenschaft gilt nicht für Volumes. Werte: `yes` oder `no`.
`mountpoint`	Zeichenkette	entf.	Legt den Einhängepunkt für das betreffende Dateisystem fest. Wenn die Eigenschaft `mountpoint` für ein Dateisystem geändert wird, werden das Dateisystem selbst und alle seine untergeordneten Dateisysteme, die den Einhängepunkt geerbt haben, ausgehängt. Wenn der neue Wert `legacy` ist, bleiben sie ausgehängt. Andernfalls werden sie, wenn die Eigenschaft vorher auf `legacy` oder `none` gesetzt war bzw. die Dateisysteme vor dem Ändern der Eigenschaft eingehängt waren, am neuen Bestimmungsort automatisch eingehängt. Darüber hinaus wird der Netzwerkzugriff auf die betreffenden Dateisysteme gesperrt und am neuen Bestimmungsort freigegeben. Weitere Informationen zur Verwendung dieser Eigenschaft finden Sie unter „Verwalten von ZFS-Einhängepunkten“ auf Seite 211.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`primarycache`	Zeichenkette	`all`	Kontrolliert, was im Primär-Cache gespeichert wird (ARC). Mögliche Werte sind `all`, `none` und `metadata`. Ist diese Eigenschaft auf `all` gesetzt, werden sowohl Benutzerdaten als auch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `none` gesetzt, werden weder Benutzerdaten noch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `metadata` gesetzt, werden nur Metadaten gespeichert.
`origin`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft für geklonte Dateisysteme bzw. Volumes, die angibt, aus welchem Snapshot der betreffende Klon erstellt wurde. Das ursprüngliche Dateisystem kann (auch mit den Optionen `-r` oder `-f`) nicht gelöscht werden, solange ein Klon vorhanden ist. Bei ungeklonten Dateisystemen besitzt diese Eigenschaft den Wert`none`.
`quota`	Zahl (oder `none`)	`none`	Beschränkt die Festplattenkapazität, die von Datasets und untergeordneten Objekten belegt werden kann. Diese Eigenschaft erzwingt einen absoluten Grenzwert der Festplattenkapazität, die belegt werden kann. Dazu zählt auch der Speicherplatz, der von untergeordneten Objekten wie Dateisystemen und Snapshots belegt wird. Das Setzen eines Kontingentes für ein untergeordnetes Objekt eines Datasets, für den bereits ein Kontingent definiert wurde, überschreibt den vom übergeordneten Dataset geerbten Wert nicht, sondern setzt darüber hinaus einen zusätzlichen Grenzwert. Kontingente können nicht für Volumes eingestellt werden, da deren Eigenschaft `volsize` bereits ein Kontingent darstellt. Weitere Informationen zum Einstellen von Kontingenten finden Sie unter „Setzen von Kontingenten für ZFS-Dateisysteme“ auf Seite 219.
`readonly`	Boolesch	`off`	Legt fest, ob ein Dataset geändert werden kann. Wenn diese Eigenschaft auf `on` gesetzt ist, können keine Änderungen vorgenommen werden. Die Abkürzung der Eigenschaft lautet `rdonly`.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`recordsize`	Zahl	`128K`	Legt eine empfohlene Blockgröße für Dateien in einem Dateisystem fest. Die Abkürzung der Eigenschaft lautet `recsize`. Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `recordsize`“ auf Seite 201.
`referenced`	Zahl	entf.	Eine schreibgeschützte Eigenschaft, die die Datenmenge festlegt, auf die ein Dataset zugreifen kann. Solche Daten können von Datasets im Pool gemeinsam genutzt oder auch nicht gemeinsam genutzt werden. Bei der Erstellung eines Snapshots bzw. Klons wird anfänglich die gleiche Festplattenkapazität referenziert, die der Kapazität des Dateisystems bzw. Snapshots entspricht, aus dem er erstellt wurde, da der Inhalt identisch ist. Die Abkürzung der Eigenschaft lautet `refer`.
`refquota`	Zahl (oder „none“)	`none`	Legt fest, wie viel Festplattenkapazität ein Dataset belegen kann. Die Eigenschaft erzwingt einen absoluten Grenzwert des belegbaren Speicherplatzes. Dieser Grenzwert umfasst keine durch untergeordnete Objekte wie z. B. Snapshots und Klone belegte Festplattenkapazität.
`refreservation`	Zahl (oder „none“)	`none`	Legt die garantierte Mindestfestplattenkapazität für ein Dataset fest (ohne untergeordnete Objekte, wie etwa Snapshots oder Klone). Liegt die belegte Festplattenkapazität unter dem hier angegebenen Wert, wird das Dataset behandelt, als würde es den mit `refreservation` angegebenen Speicherplatz belegen. Reservierungen durch `refreservation` werden in die Berechnung der Festplattenkapazität für das diesem Dataset übergeordnete Dataset einbezogen und auf die Kontingente und Reservierung für das übergeordnete Dataset angerechnet. Wenn `refreservation` gesetzt ist, wird ein Snapshot nur zugelassen, wenn außerhalb dieser Reservierung genügend freier Speicherplatz im Pool vorhanden ist, um die Menge der aktuell *referenzierten* Byte im Dataset aufzunehmen. Die Abkürzung der Eigenschaft lautet `refreserv`.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`reservation`	Zahl (oder „none“)	`none`	Legt die Mindestfestplattenkapazität für ein Dataset und seine untergeordneten Objekte fest. Liegt die belegte Festplattenkapazität unter dem hier angegebenen Wert, wird das Dataset behandelt, als würde es den in dieser Reservierung angegebenen Speicherplatz belegen. Reservierungen werden in die Berechnung der Festplattenkapazität für das diesem Dataset übergeordneten Dataset einbezogen und auf die Kontingente und Reservierung für das übergeordneten Dataset angerechnet. Die Abkürzung der Eigenschaft lautet `reserv`. Weitere Informationen dazu finden Sie unter „Setzen von Reservierungen für ZFS-Dateisysteme“ auf Seite 222.
`secondarycache`	Zeichenkette	`all`	Kontrolliert, was im Sekundär-Cache gespeichert wird (L2ARC). Mögliche Werte sind `all`, `none` und `metadata`. Ist diese Eigenschaft auf `all` gesetzt, werden sowohl Benutzerdaten als auch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `none` gesetzt, werden weder Benutzerdaten noch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `metadata` gesetzt, werden nur Metadaten gespeichert.
`setuid`	Boolesch	`on`	Legt fest, ob das `setuid`-Bit in einem Dateisystem berücksichtigt wird.
`shareiscsi`	Zeichenkette	off	Legt fest, ob ein ZFS-Volume gemeinsam als iSCSI-Ziel genutzt wird. Eigenschaftswerte: `on`, `off` und `type=disk`. Sie können `shareiscsi=on` für ein Dateisystem setzen, sodass alle ZFS-Volumes innerhalb des Dateisystems standardmäßig freigegeben sind. Das Setzen dieser Eigenschaft für ein Dateisystem hat jedoch keine direkte Auswirkung.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`sharenfs`	Zeichenkette	`off`	Legt fest, ob das Dateisystem über NFS zugänglich ist und welche Optionen verwendet werden. Wenn diese Eigenschaft auf `on` gesetzt ist, wird der Befehl `zfs share` ohne Optionen aufgerufen. Andernfalls wird der Befehl `zfs share` mit den Optionen aufgerufen, die dem Inhalt dieser Eigenschaft entsprechen. Wenn die Eigenschaft auf `off` gesetzt ist, wird das Dateisystem über die älteren Befehle `share` und `unshare` und die Datei `dfstab` verwaltet. Weitere Informationen zur Nutzung von ZFS-Dateisystemen für den Netzwerkzugang finden Sie unter „Freigeben und Sperren von ZFS-Dateisystemen" auf Seite 216.
`snapdir`	Zeichenkette	`hidden`	Legt fest, ob das Verzeichnis `.zfs` in der Root des Dateisystems verborgen oder sichtbar ist. Weitere Informationen zur Verwendung von Snapshots finden Sie unter „Überblick über ZFS-Snapshots" auf Seite 225.
`type`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft, die den Typ des betreffenden Datasets (`filesystem` (Dateisystem oder Klon), `volume` oder `snapshot`) angibt.
`used`	Zahl	entf.	Schreibgeschützte Eigenschaft, die die für ein Dataset und alle seine untergeordneten Objekte belegte Festplattenkapazität angibt. Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `used`" auf Seite 199.
`usedbychildren`	Zahl	`off`	Schreibgeschützte Eigenschaft, die die Festplattenkapazität angibt, die von untergeordneten Objekten dieses Datasets beansprucht wird und die beim Löschen dieser untergeordneten Objekte frei werden würde. Die Abkürzung für die Eigenschaft ist `usedchild`.
`usedbydataset`	Zahl	`off`	Schreibgeschützte Eigenschaft, die die Festplattenkapazität angibt, die vom Dataset selbst beansprucht wird und die beim Löschen des Datasets und vorherigem Löschen aller Snapshots und Entfernen von `refreservation` frei werden würde. Die Abkürzung für diese Eigenschaft ist `usedds`.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`usedbyrefreservation`	Zahl	`off`	Schreibgeschützte Eigenschaft, die die von einem `refreservation`-Set auf diesem Dataset beanspruchte Festplattenkapazität angibt, die beim Entfernen von `refreservation` frei werden würde. Die Abkürzung für die Eigenschaft ist `usedds`.
`usedbysnapshots`	Zahl	`off`	Schreibgeschützte Eigenschaft, die die von Snapshots dieses Datasets beanspruchte Festplattenkapazität angibt. Insbesondere geht es dabei um die Festplattenkapazität, die beim Löschen aller Snapshots des Datasets frei werden würde. Beachten Sie, dass es sich bei diesem Wert nicht einfach um die Summe der `used`-Eigenschaften der Snapshots handelt, da Speicherplatz von mehreren Snapshots gemeinsam genutzt werden kann. Die Abkürzung für diese Eigenschaft ist `usedsnap`.
`version`	Zahl	entf.	Die aktuelle, von der Pool-Version unabhängige Version eines Dateisystems auf der Festplatte. Diese Eigenschaft kann nur bei einer nachfolgenden Version der unterstützten Software gesetzt werden. Weitere Informationen finden Sie unter `zfs upgrade`-Befehl.
`volsize`	Zahl	entf.	Legt die logische Größe eines Volumes fest (gilt nur für Volumes). Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `volsize`“ auf Seite 202.
`volblocksize`	Zahl	`8 kB`	Legt die Blockgröße eines Volumes fest (gilt nur für Volumes). Nach dem Schreiben von Daten auf das betreffende Volume kann die Blockgröße nicht mehr geändert werden. Deswegen muss diese Eigenschaft bei der Erstellung des Volumes gesetzt werden. Die Standardblockgröße für Volumes ist 8 KB. Es sind alle Werte zur Potenz von 2 im Bereich von 512 Byte bis 128 KB zulässig. Die Abkürzung der Eigenschaft lautet `volblock`.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`zoned`	Boolesch	entf.	Gibt an, ob dieses Dataset zu einer nicht-globalen Zone hinzugefügt wurde. Wenn diese Eigenschaft gesetzt ist, befindet sich der Einhängepunkt nicht in der globalen Zone, und ZFS kann ein solches Dateisystem auf Anforderung nicht einhängen. Bei der ersten Installation einer Zone wird diese Eigenschaft für alle hinzugefügten Dateisysteme gesetzt. Weitere Informationen zur Verwendung von ZFS mit installierten Zonen finden Sie in „Verwendung von ZFS in einem Solaris-System mit installierten Zonen“ auf Seite 290.
`xattr`	Boolesch	`on`	Legt fest, ob für das betreffende Dateisystem erweiterte Attribute aktiviert (`on`) oder deaktiviert sind (`off`).

Schreibgeschützte native ZFS-Eigenschaften

Schreibgeschützte native Eigenschaften können gelesen, aber nicht gesetzt werden. Schreibgeschützte native Eigenschaften werden nicht vererbt. Einige native Eigenschaften gelten nur für bestimmte Dataset-Typen. In solchen Fällen ist der entsprechende Dataset-Typ in der Beschreibung in Tabelle 6–1 aufgeführt.

Die schreibgeschützten nativen Eigenschaften sind hier aufgeführt und in Tabelle 6–1 beschrieben.

- `available`
- `compressratio`
- `creation`
- `mounted`
- `origin`
- `referenced`
- `type`
- `used`

  Ausführliche Informationen finden Sie unter „Die Eigenschaft `used`“ auf Seite 199.
- `usedbychildren`
- `usedbydataset`
- `usedbyrefreservation`
- `usedbysnapshots`

Weitere Informationen zur Berechnung von Festplattenkapazität (einschließlich Informationen zu den Eigenschaften `used`, `referenced` und `available`) finden Sie unter „Berechnung von ZFS-Festplattenkapazität“ auf Seite 62.

Die Eigenschaft `used`

Die Eigenschaft `used` ist eine schreibgeschützte Eigenschaft, die die für ein Dataset und alle seine untergeordneten Objekte belegte Festplattenkapazität angibt. Dieser Wert wird gegen die für das Dataset gesetzten Kontingente und Reservierungen geprüft. Die belegte Festplattenkapazität enthält nicht die Reservierungen für das Dataset selbst, schließt jedoch die Reservierungen für untergeordnete Datasets ein. Die Festplattenkapazität, die das Dataset aufgrund seines übergeordneten Datasets belegt, sowie die Festplattenkapazität, die beim rekursiven Löschen des Datasets freigegeben wird, sind größer als die Kapazität, die das Dataset für den belegten Speicherplatz und die Reservierung benötigt.

Bei der Erstellung von Snapshots wird diese Festplattenkapazität anfänglich vom Snapshot und dem Dateisystem (sowie eventuellen früheren Snapshots) gemeinsam genutzt. Wenn sich ein Dateisystem mit der Zeit ändert, gehört zuvor gemeinsam genutzte Festplattenkapazität dann ausschließlich dem Snapshot und wird in die Berechnung des vom Snapshot belegten Speicherplatzes einbezogen. Wie viel Festplattenkapazität von einem Snapshot belegt wird, hängt von den speziellen Daten des Snapshots ab. Zudem kann durch das Löschen von Snapshots die Festplattenkapazität, die Snapshots eindeutig zugewiesen ist (und von diesen verwendet wird), größer werden. Weitere Informationen zu Snapshots und Speicherplatzaspekten finden Sie in „Verhalten bei ungenügendem Speicherplatz“ auf Seite 63.

In die belegte, verfügbare und referenzierte Festplattenkapazität werden keine anstehenden Änderungen einbezogen. Anstehende Änderungen werden im Allgemeinen innerhalb weniger Sekunden abgeschlossen. Nach Abschluss eines Schreibvorgangs auf dem Datenträger durch die Funktion `fsync(3c)` oder `O_SYNC` werden die Informationen zur belegten Festplattenkapazität nicht unbedingt sofort aktualisiert.

Die Informationen der Eigenschaften `usedbychildren`, `usedbydataset` , `usedbyrefreservation` und `usedbysnapshots` kann mit dem Befehl `zfs list - o space` angezeigt werden. Diese Eigenschaften bestimmen die Eigenschaft `used` für Festplattenkapazität, die von untergeordneten Objekten belegt wird. Weitere Informationen finden Sie in Tabelle 6–1.

Konfigurierbare native ZFS-Eigenschaften

Konfigurierbare native Eigenschaften können gelesen und gesetzt werden. Konfigurierbare native Eigenschaften werden mithilfe des Befehls `zfs set`(siehe „Setzen von ZFS-Eigenschaften“ auf Seite 206) bzw. `zfs create` (siehe „Erstellen eines ZFS-Dateisystems“ auf Seite 186) gesetzt. Außer Kontingenten und Reservierungen werden konfigurierbare Eigenschaften vererbt. Weitere Informationen zu Kontingenten und Reservierungen finden Sie unter „Einstellen von ZFS-Kontingenten und -Reservierungen“ auf Seite 218.

Einige konfigurierbare native Eigenschaften gelten nur für bestimmte Dataset-Typen. In solchen Fällen ist der entsprechende Dataset-Typ in der Beschreibung in Tabelle 6–1 aufgeführt. Sofern nichts Anderes vermerkt ist, gilt eine Eigenschaft für alle Dataset-Typen: Dateisysteme, Volumes, Klone und Snapshots.

Die konfigurierbaren Eigenschaften sind hier aufgeführt und in Tabelle 6–1 beschrieben.

- `aclinherit`

  Eine ausführliche Beschreibung finden Sie unter „Eigenschaften von Zugriffssteuerungslisten" auf Seite 249.
- `aclmode`

  Eine ausführliche Beschreibung finden Sie unter „Eigenschaften von Zugriffssteuerungslisten" auf Seite 249.
- `atime`
- `canmount`
- `checksum`
- `compression`
- `copies`
- `devices`
- `exec`
- `mountpoint`
- `primarycache`
- `quota`
- `readonly`
- `recordsize`

  Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `recordsize`" auf Seite 201.
- `refquota`
- `refreservation`
- `reservation`
- `secondarycache`
- `shareiscsi`
- `sharenfs`
- `setuid`
- `snapdir`
- `version`
- `volsize`

Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `volsize`“ auf Seite 202.

- `volblocksize`
- `zoned`
- `xattr`

Die Eigenschaft `canmount`

Wenn die Eigenschaft `canmount` auf `off` gesetzt wird, kann das Dateisystem nicht mithilfe der Befehle `zfs mount` bzw. `zfs mount -a` eingehängt werden. Das Setzen dieser Eigenschaft auf `off` gleicht dem Setzen der Eigenschaft `mountpoint` auf `none`. Der Unterschied besteht darin, dass das Dataset trotzdem noch die normale Eigenschaft `mountpoint` besitzt, die vererbt werden kann. Sie können diese Eigenschaft beispielsweise auf `off` setzen und Eigenschaften setzen, die an untergeordnete Dateisysteme vererbt werden. Das übergeordnete Dateisystem selbst wird jedoch nicht eingehängt und ist nicht für Benutzer zugänglich. In diesem Fall dient das übergeordnete Dateisystem als *Container*, für den Sie Eigenschaften festlegen können. Der Container selbst ist jedoch nicht zugänglich.

Im folgenden Beispiel wird das Dateisystem `userpool` erstellt und dessen Eigenschaft `canmount` auf `off` gesetzt. Einhängepunkte für untergeordnete Benutzerdateisysteme werden auf ein einziges Verzeichnis (`/export/home`) gesetzt. Am übergeordneten Dateisystem gesetzte Eigenschaften werden von untergeordneten Dateisystemen geerbt; das übergeordnete Dateisystem selbst wird jedoch nicht eingehängt.

```
# zpool create userpool mirror c0t5d0 c1t6d0
# zfs set canmount=off userpool
# zfs set mountpoint=/export/home userpool
# zfs set compression=on userpool
# zfs create userpool/user1
# zfs create userpool/user2
# zfs mount
userpool/user1                  /export/home/user1
userpool/user2                  /export/home/user2
```

Durch Setzen der Eigenschaft `canmount` auf `noauto` kann das Dataset nur ausdrücklich, nicht aber automatisch eingehängt werden. Über diese Einstellung wird von der Oracle Solaris Upgrade-Software festgelegt, dass nur Datasets aus der aktiven Boot-Umgebung beim Booten eingehängt werden.

Die Eigenschaft `recordsize`

Die Eigenschaft `recordsize` legt eine empfohlene Blockgröße für Dateien im Dateisystem fest.

Diese Eigenschaft dient zur Zusammenarbeit mit Datenbanken, die auf Dateien in festen Blockgrößen zugreifen. ZFS passt Blockgrößen automatisch an interne Algorithmen an, die für typische Zugriffsstrukturen optimiert wurden. Für Datenbanken, die sehr große Dateien erstellen, Dateien jedoch mit wahlfreiem Zugriff in kleineren Blöcken lesen, können diese Algorithmen unter Umständen nicht optimal sein. Wenn Sie die Eigenschaft `recordsize` auf

einen Wert setzen, der der Datensatzgröße der betreffenden Datenbank entspricht bzw. größer als diese ist, kann die Leistung bedeutend verbessert werden. Die Verwendung dieser Eigenschaft für allgemeine Dateisysteme kann sich negativ auf die Leistung auswirken und wird nicht empfohlen. Die angegebene Größe muss ein Zweierpotenzwert sein, der größer als oder gleich 512 Byte und kleiner als oder gleich 128 KB ist. Das Ändern der Eigenschaft `recordsize` eines Dateisystems wirkt sich nur auf Dateien aus, die nach der Änderung erstellt wurden. Bereits vorhandene Dateien bleiben unverändert.

Die Abkürzung der Eigenschaft lautet `recsize`.

Die Eigenschaft `volsize`

Die Eigenschaft `volsize` legt die logische Größe eines Volumes fest. Standardmäßig wird beim Erstellen eines Volumes Speicherplatz der gleichen Kapazität reserviert. Alle Änderungen der Eigenschaft `volsize` werden entsprechend in der Reservierung geändert. Diese Überprüfungen dienen zum Verhindern unerwarteten Systemverhaltens. Je nachdem, wie ein Volume benutzt wird, kann es undefiniertes Systemverhalten oder Datenbeschädigung verursachen, wenn es weniger Speicherplatz enthält, als es eigentlich angefordert hat. Solche Effekte können auch auftreten, wenn die Kapazität eines Volumes geändert wird, während es in Benutzung ist. Dies gilt besonders dann, wenn die Volume-Kapazität verkleinert wird. Gehen Sie bei der Änderung einer Volume-Kapazität stets mit äußerster Sorgfalt vor.

Obwohl dies nicht empfohlen wird, können Sie ein Sparse-Volume erstellen, indem Sie für den Befehl `-zfs create -V` das Flag `s` angeben oder die Reservierung nach der Erstellung des Volumes entsprechend ändern. Bei *Sparse-Volumes* ist der Wert der Reservierung ungleich der Volume-Kapazität. Änderungen der Eigenschaft `volsize` wirken sich bei Sparse-Volumes nicht auf den Reservierungswert aus.

Weitere Informationen zur Verwendung von Volumes finden Sie unter „ZFS-Volumes“ auf Seite 287.

Benutzerdefinierte ZFS-Eigenschaften

Zusätzlich zu den nativen Eigenschaften unterstützt ZFS auch beliebige benutzerdefinierte Eigenschaften. Benutzerdefinierte Eigenschaften wirken sich nicht auf das ZFS-Verhalten aus, können jedoch zum Versehen von Datasets mit Informationen, die für Ihre lokalen Gegebenheiten wichtig sind, verwendet werden.

Namen benutzerdefinierter Eigenschaften müssen den folgenden Konventionen genügen:

- Sie müssen einen Doppelpunkt (:) enthalten, damit sie von nativen Eigenschaften unterschieden werden können.
- Sie dürfen Kleinbuchstaben, Zahlen und die folgenden Interpunktionszeichen enthalten: ':', '+', '.', '_'.
- Der Name einer benutzerdefinierten Eigenschaft darf maximal 256 Zeichen lang sein.

Namen benutzerdefinierter Eigenschaften sollten generell in die folgenden beiden Komponenten aufgeteilt werden, obwohl dies für ZFS nicht obligatorisch ist:

module:property

Bei der programmatischen Verwendung von Eigenschaften sollten Sie für die Komponente *Modul* einen umgekehrten DNS-Domänennamen verwenden. Dadurch wird die Wahrscheinlichkeit verringert, dass zwei unabhängig voneinander entwickelte Pakete die gleiche Eigenschaft für unterschiedliche Zwecke nutzen. Eigenschaftsnamen, die mit `com.sun` beginnen, sind für Oracle Corporation reserviert.

Die Werte benutzerdefinierter Eigenschaften müssen folgenden Konventionen entsprechen:

- Sie müssen aus beliebigen Zeichenketten bestehen, die immer vererbt und niemals validiert werden.
- Der Wert einer benutzerdefinierten Eigenschaft darf maximal 1024 Zeichen lang sein.

Beispiel:

```
# zfs set dept:users=finance userpool/user1
# zfs set dept:users=general userpool/user2
# zfs set dept:users=itops userpool/user3
```

Alle Befehle, die Eigenschaften verwenden (z. B. `zfs list`, `zfs get`, `zfs set` usw.) können native und benutzerdefinierte Eigenschaften nutzen.

Beispiel:

```
zfs get -r dept:users userpool
NAME            PROPERTY    VALUE           SOURCE
userpool        dept:users  all             local
userpool/user1  dept:users  finance         local
userpool/user2  dept:users  general         local
userpool/user3  dept:users  itops           local
```

Benutzerdefinierte Eigenschaften können mit dem Befehl `zfs inherit` gelöscht werden. Beispiel:

```
# zfs inherit -r dept:users userpool
```

Wenn die betreffende Eigenschaft nicht in einem übergeordneten Dataset definiert wurde, wird sie komplett entfernt.

Abfragen von ZFS-Dateisysteminformationen

Der Befehl `zfs list` bietet einen umfassenden Mechanismus zum Anzeigen und Abfragen von Dataset-Informationen. In diesem Abschnitt werden grundlegende und komplexere Abfragen erläutert.

Auflisten grundlegender ZFS-Informationen

Mit dem Befehl `zfs list` ohne Optionen können Sie sich grundlegende Dataset-Informationen anzeigen lassen. Dieser Befehl zeigt die Namen aller Datasets im System sowie die Werte der Eigenschaften `used`, `available`, `referenced` und `mountpoint` an. Weitere Informationen zu diesen Eigenschaften finden Sie unter „ZFS-Eigenschaften“ auf Seite 189.

Beispiel:

```
# zfs list
NAME                    USED  AVAIL  REFER  MOUNTPOINT
pool                    476K  16.5G    21K  /pool
pool/clone               18K  16.5G    18K  /pool/clone
pool/home               296K  16.5G    19K  /pool/home
pool/home/marks         277K  16.5G   277K  /pool/home/marks
pool/home/marks@snap       0      -   277K  -
pool/test                18K  16.5G    18K  /test
```

Mithilfe dieses Befehls können Sie auch Informationen zu bestimmten Datasets anzeigen. Geben Sie dazu in der Befehlszeile den Namen des gewünschten Datasets an. Darüber hinaus können Sie mit der Option `-r` rekursiv Informationen zu allen untergeordneten Datasets anzeigen. Beispiel:

```
# zfs list -r pool/home/marks
NAME                    USED  AVAIL  REFER  MOUNTPOINT
pool/home/marks         277K  16.5G   277K  /pool/home/marks
pool/home/marks@snap       0      -   277K  -
```

Sie können den Befehl `zfs list` zusammen mit dem Einhängepunkt eines Dateisystems verwenden. Beispiel:

```
# zfs list /pool/home/marks
NAME               USED  AVAIL  REFER  MOUNTPOINT
pool/home/marks    277K  16.5G   277K  /pool/home/marks
```

Das folgende Beispiel zeigt, wie grundlegende Informationen zum Dateisystem `tank/home/chua` und allen seinen untergeordneten Datasets angezeigt werden können:

```
# zfs list -r tank/home/chua
NAME                         USED  AVAIL  REFER  MOUNTPOINT
tank/home/chua                   26.0K  4.81G  10.0K  /tank/home/chua
tank/home/chua/projects       16K  4.81G   9.0K  /tank/home/chua/projects
tank/home/chua/projects/fs1    8K  4.81G     8K  /tank/home/chua/projects/fs1
tank/home/chua/projects/fs2    8K  4.81G     8K  /tank/home/chua/projects/fs2
```

Zusätzliche Informationen zum Befehl `zfs list` finden Sie in der Man Page `zfs`(1M).

Erstellen komplexer ZFS-Abfragen

Die Ausgabe des Befehls `zfs list` kann mithilfe der Optionen `-o`, `-f` und `-H` speziell angepasst werden.

Mit der Option `-o` und einer kommagetrennten Liste gewünschter Eigenschaften können Sie die Ausgabe von Eigenschaftswerten anpassen. Sie können jede Dataset-Eigenschaft als gültiges Argument angeben. Eine Liste aller unterstützten Dataset-Eigenschaften finden Sie unter „ZFS-Eigenschaften“ auf Seite 189. Zusätzlich zu den hier definierten Eigenschaften kann die Option `-o` auch das Literal `name` enthalten. In diesem Fall enthält die Befehlsausgabe auch den Namen des Datasets.

Im folgenden Beispiel wird mithilfe von `zfs list` der Dataset-Name zusammen mit den Eigenschaftswerten `sharenfs` und `mountpoint` angezeigt.

```
# zfs list -o name,sharenfs,mountpoint
NAME                   SHARENFS         MOUNTPOINT
tank                   off              /tank
tank/home              on               /tank/home
tank/home/ahrens       on               /tank/home/ahrens
tank/home/bonwick      on               /tank/home/bonwick
tank/home/chua         on               /tank/home/chua
tank/home/eschrock     on               legacy
tank/home/moore        on               /tank/home/moore
tank/home/tabriz       ro               /tank/home/tabriz
```

Mit der Option `-t` können Sie festlegen, welche Dataset-Typen angezeigt werden sollen. Zulässige Typen sind in der folgenden Tabelle aufgeführt.

TABELLE 6–2 ZFS-Dataset-Typen

Typ	Beschreibung
`filesystem`	Dateisysteme und Klone
`volume`	Volumes
`snapshot`	Snapshots

Die Option `-t` liest eine kommagetrennte Liste der anzuzeigenden Dataset-Typen. Im folgenden Beispiel wird mithilfe der Optionen `-t` und `- o` für alle Dateisysteme gleichzeitig der Name und die Eigenschaft `used` angezeigt:

```
# zfs list -t filesystem -o name,used
NAME              USED
pool              476K
```

```
pool/clone         18K
pool/home         296K
pool/home/marks   277K
pool/test          18K
```

Mithilfe der Option `-H` kann bei der Ausgabe des Befehls `zfs list` die Titelzeile unterdrückt werden. Bei Verwendung der Option `-H` werden Leerzeichen durch Tabulatorzeichen ersetzt. Diese Option ist bei der Verwendung der Befehlsausgabe für programmatische Anwendungen (z. B. Skripten) nützlich. Das folgende Beispiel zeigt die Ausgabe des Befehls `zfs list` mit der Option `-H`.

```
# zfs list -H -o name
pool
pool/clone
pool/home
pool/home/marks
pool/home/marks@snap
pool/test
```

Verwalten von ZFS-Eigenschaften

Dataset-Eigenschaften werden mithilfe der Unterbefehle `set`, `inherit` und `get` des Befehls `zfs` verwaltet.

- „Setzen von ZFS-Eigenschaften" auf Seite 206
- „Vererben von ZFS-Eigenschaften" auf Seite 207
- „Abfragen von ZFS-Eigenschaften" auf Seite 208

Setzen von ZFS-Eigenschaften

Sie können konfigurierbare Dataset-Eigenschaften mit dem Befehl `zfs set` setzen. Bei der Erstellung eines Datasets können Eigenschaften auch mit dem Befehl `zfs create` gesetzt werden. Eine Liste der konfigurierbaren Dataset-Eigenschaften finden Sie unter „Konfigurierbare native ZFS-Eigenschaften" auf Seite 199.

Der Befehl `zfs set` verwendet ein Eigenschaft-Wert-Paar im Format *Eigenschaft*=*Wert*, dem ein Dataset-Name folgt. Während eines Aufrufs von `zfs set` kann nur eine Eigenschaft gesetzt oder geändert werden.

Im folgenden Beispiel wird die Eigenschaft `atime` von `tank/home` auf `off` gesetzt.

```
# zfs set atime=off tank/home
```

Darüber hinaus können bei der Erstellung eines Dateisystems beliebige Dateisystemeigenschaften gesetzt werden. Beispiel:

```
# zfs create -o atime=off tank/home
```

Spezielle nummerische Eigenschaftswerte können durch Verwendung der folgenden verständlichen Suffixe (in ansteigender Größenordnung) angegeben werden: BKMGTPEZ. Allen diesen Suffixen außer dem Suffix B, das für Byte steht, kann ein b (für „Byte") nachgestellt werden. In den folgenden vier Beispielen des Befehls zfs set werden entsprechende nummerische Ausdrücke angegeben, mit denen die Eigenschaft quota gesetzt wird. Damit werden Kontingente im Dateisystem tank/home/marks auf 50 GB gesetzt:

```
# zfs set quota=50G tank/home/marks
# zfs set quota=50g tank/home/marks
# zfs set quota=50GB tank/home/marks
# zfs set quota=50gb tank/home/marks
```

Bei Zeichenkettenwerten wird Groß- und Kleinschreibung unterschieden. Diese Werte dürfen nur Kleinbuchstaben enthalten. Ausnahmen bilden die Werte der Eigenschaften mountpoint und sharenfs; die Werte dieser Eigenschaften dürfen sowohl Groß- als auch Kleinbuchstaben enthalten.

Weitere Informationen zum Befehl zfs set finden Sie in der Man Page zfs(1M).

Vererben von ZFS-Eigenschaften

Alle konfigurierbaren Eigenschaften mit der Ausnahme von Kontingenten und Reservierungen erben ihren Wert von ihrem übergeordneten Dataset, es sei denn, diese Werte sind im untergeordneten Dataset explizit gesetzt. Wenn das entsprechende übergeordnete Dateisystem für eine vererbte Eigenschaft keinen Wert besitzt, wird der Standardwert für die betreffende Eigenschaft verwendet. Mit dem Befehl zfs inherit können Sie Eigenschaftswerte zurücksetzen, was zur Folge hat, dass der vom übergeordneten Dateisystem vererbte Wert verwendet wird.

Im folgenden Beispiel wird mithilfe des Befehls zfs set die Komprimierung für das Dateisystem tank/home/bonwick aktiviert. Danach wird zfs inherit verwendet, um die Eigenschaft compression zu löschen, wodurch die Eigenschaft den Standardwert (off) des erbt. Da weder bei home noch bei tank der Wert der Eigenschaft compression lokal gesetzt wurde, wird der Standardwert verwendet. Wäre bei beiden die Komprimierung aktiviert, würde der Wert des direkten übergeordneten Dateisystems (in diesem Beispiel home) verwendet werden.

```
# zfs set compression=on tank/home/bonwick
# zfs get -r compression tank
NAME              PROPERTY      VALUE                     SOURCE
tank              compression   off                       default
tank/home         compression   off                       default
tank/home/bonwick compression   on                        local
# zfs inherit compression tank/home/bonwick
# zfs get -r compression tank
NAME              PROPERTY      VALUE                     SOURCE
tank              compression   off                       default
```

```
tank/home        compression   off                        default
tank/home/bonwick compression  off                        default
```

Der Unterbefehl `inherit` wird bei Angabe der Option `-r` rekursiv ausgeführt. Im folgenden Beispiel wird durch den Befehl der Wert für die Eigenschaft `compression` von `tank/home` und allen eventuell vorhandenen untergeordneten Dateisystemen geerbt:

```
# zfs inherit -r compression tank/home
```

Hinweis – Bitte beachten Sie, dass die Option `-r` die Eigenschaftswerte aller untergeordneten Datasets zurücksetzt.

Weitere Informationen zum Befehl `zfs inherit` finden Sie in der Man Page `zfs`(1M).

Abfragen von ZFS-Eigenschaften

Am Einfachsten können Eigenschaftswerte mit dem Befehl `zfs list` abgefragt werden. Weitere Informationen dazu finden Sie unter „Auflisten grundlegender ZFS-Informationen“ auf Seite 204. Für komplexere Abfragen und Skripten sollten Sie den Befehl `zfs get` verwenden, da dieser ausführlichere Informationen in einem anpassbaren Format anzeigt.

Sie können Dataset-Eigenschaften mit dem Befehl `zfs get` abrufen. Das folgende Beispiel zeigt, wie ein Eigenschaftswert eines Datasets abgerufen werden kann:

```
# zfs get checksum tank/ws
NAME             PROPERTY       VALUE                      SOURCE
tank/ws          checksum       on                         default
```

In der vierten Spalte `SOURCE` wird der Ursprung des betreffende Eigenschaftswerts angezeigt. In der folgenden Tabelle werden die möglichen Ursprungswerte erläutert.

TABELLE 6–3 Mögliche `SOURCE`-Werte (Befehl `zfs get`)

SOURCE-Wert	Beschreibung
`default`	Dieser Eigenschaftswert wurde für dieses Dataset bzw. seine übergeordneten Datasets nie explizit gesetzt. Es wird der Standardwert für diese Eigenschaft verwendet.
`inherited from` *Dataset-Name*	Dieser Eigenschaftswert wurde vom übergeordneten Dataset geerbt, das in *dataset-name* angegeben ist.
`local`	Dieser Eigenschaftswert wurde mithilfe von `zfs set` für dieses Dataset explizit gesetzt.

TABELLE 6–3 Mögliche SOURCE-Werte (Befehl `zfs get`) *(Fortsetzung)*

SOURCE-Wert	Beschreibung
`temporary`	Dieser Eigenschaftswert wurde mithilfe von `zfs mount - o` gesetzt und gilt nur solange, wie das Dateisystem eingehängt ist. Weitere Informationen zu temporären Eigenschaften von Einhängepunkten finden Sie unter „Verwenden temporärer Einhängepunkte" auf Seite 215.
- (keiner)	Diese Eigenschaft ist schreibgeschützt. Ihr Wert wird von ZFS bereitgestellt.

Sie können alle Dataset-Eigenschaftswerte mit dem speziellen Schlüsselwort `all` abrufen. In den folgenden Beispielen wird das Schlüsselwort `all` verwendet:

```
# zfs get all tank/home
NAME       PROPERTY         VALUE                  SOURCE
tank/home  type             filesystem             -
tank/home  creation         Tue Jun 29 11:44 2010  -
tank/home  used             21K                    -
tank/home  available        66.9G                  -
tank/home  referenced       21K                    -
tank/home  compressratio    1.00x                  -
tank/home  mounted          yes                    -
tank/home  quota            none                   default
tank/home  reservation      none                   default
tank/home  recordsize       128K                   default
tank/home  mountpoint       /tank/home             default
tank/home  sharenfs         off                    default
tank/home  checksum         on                     default
tank/home  compression      off                    default
tank/home  atime            on                     default
tank/home  devices          on                     default
tank/home  exec             on                     default
tank/home  setuid           on                     default
tank/home  readonly         off                    default
tank/home  zoned            off                    default
tank/home  snapdir          hidden                 default
tank/home  aclmode          groupmask              default
tank/home  aclinherit       restricted             default
tank/home  canmount         on                     default
tank/home  shareiscsi       off                    default
tank/home  xattr            on                     default
tank/home  copies           1                      default
tank/home  version          4                      -
tank/home  utf8only         off                    -
tank/home  normalization    none                   -
tank/home  casesensitivity  sensitive              -
tank/home  vscan            off                    default
tank/home  nbmand           off                    default
tank/home  sharesmb         off                    default
tank/home  refquota         none                   default
tank/home  refreservation   none                   default
tank/home  primarycache     all                    default
tank/home  secondarycache   all                    default
```

```
tank/home   usedbysnapshots       0                       -
tank/home   usedbydataset         21K                     -
tank/home   usedbychildren        0                       -
tank/home   usedbyrefreservation  0                       -
tank/home   logbias               latency                 default
```

Hinweis – Die Eigenschaften `casesensitivity`, `nbmand`, `normalization`, `sharesmb`, `utf8only` und `vscan` sind in Oracle Solaris 10 nicht voll funktionsfähig, da der Oracle Solaris SMB-Service nicht von Oracle Solaris 10 unterstützt wird.

Mit der Option `-s` des Befehls `zfs get` können Sie die anzuzeigenden Eigenschaften nach Ursprungstyp angeben. Diese Option liest eine kommagetrennte Liste der gewünschten Ursprungstypen ein. Es werden nur Eigenschaften des gewünschten Ursprungstyps angezeigt. Zulässige Ursprungstypen sind `local`, `default`, `inherited`, `temporary` und `none`. Das folgende Beispiel zeigt alle Eigenschaften, die in `pool` lokal gesetzt wurden.

```
# zfs get -s local all pool
NAME              PROPERTY      VALUE                          SOURCE
pool              compression   on                             local
```

Alle der o. g. Optionen können zusammen mit der Option `-r` verwendet werden, um die angegebenen Eigenschaften aller untergeordneten Datasets rekursiv anzuzeigen. Im folgenden Beispiel werden alle temporären Eigenschaften aller Datasets in `tank` rekursiv angezeigt:

```
# zfs get -r -s temporary all tank
NAME              PROPERTY       VALUE                          SOURCE
tank/home           atime          off                          temporary
tank/home/bonwick   atime          off                          temporary
tank/home/marks     atime          off                          temporary
```

Mithilfe des Befehls `zfs get` können Sie Eigenschaftswerte abfragen ohne ein Zieldateisystem anzugeben, was bedeutet, dass alle Pools bzw. Dateisysteme abgefragt werden. Beispiel:

```
# zfs get -s local all
tank/home               atime          off                     local
tank/home/bonwick       atime          off                     local
tank/home/marks         quota          50G                     local
```

Weitere Informationen zum Befehl `zfs get` finden Sie in der Man Page `zfs`(1M).

Abfragen von ZFS-Eigenschaften für Skripten

Der Befehl `zfs get` unterstützt die Optionen `-H` und `- o`, die speziell für die Verwendung dieses Befehl in Skripten vorgesehen sind. Sie können die Option `-H` verwenden, um die Kopfzeileninformationen zu unterdrücken und Leerzeichen durch Tabulatorzeichen zu ersetzen. Dadurch können Daten einfach analysiert werden. Sie können die Option `-o` verwenden, um die Ausgabe wie folgt anzupassen:

- Das Literal `name` kann zusammen mit einer kommagetrennten Liste von Eigenschaften verwendet werden (siehe Abschnitt „ZFS-Eigenschaften“ auf Seite 189.

- Eine kommagetrennte Liste von Literalfeldern, `name`, `value`, `property` und `source`, wird ausgegeben, der ein Leerzeichen und ein Argument folgt. Diese Liste ist eine kommagetrennte Liste von Eigenschaften.

Das folgende Beispiel zeigt, wie mithilfe der Optionen `-H` und `-o` des Befehls `zfs get` ein einzelner Wert abgerufen werden kann.

```
# zfs get -H -o value compression tank/home
on
```

Die Option `-p` gibt nummerische Werte exakt aus. 1 MB wird beispielsweise als 1000000 ausgegeben. Diese Option lässt sich wie folgt verwenden:

```
# zfs get -H -o value -p used tank/home
182983742
```

Mit der Option `-r` und allen der o. g. Optionen können Sie Werte für alle untergeordneten Datasets rekursiv abrufen. Im folgenden Beispiel werden die Optionen `-H`, `-o` und `- r` verwendet, um den Dataset-Namen sowie der Wert der Eigenschaft `used` für `export/home` und die untergeordneten Objekte abzurufen, während die Kopfzeile der Befehlsausgabe unterdrückt wird:

```
# zfs get -H -o name,value -r used export/home
export/home             5.57G
export/home/marks       1.43G
export/home/maybee      2.15G
```

Einhängen und Freigeben von ZFS-Dateisystemen

In diesem Abschnitt wird die Verwaltung von Einhängepunkten und freigegebenen Dateisystemen in ZFS beschrieben.

Verwalten von ZFS-Einhängepunkten

Ein ZFS-Dateisystem wird automatisch eingehängt, wenn es erstellt wird. In diesem Abschnitt wird beschrieben, wie Sie das Verhalten eines Einhängepunkts für ein Dateisystem bestimmen können.

Sie können den Standard-Einhängepunkt für ein Pool-Dataset bei dessen Erstellung auch mithilfe der Option `m` von `-zpool create` setzen. Weitere Informationen zum Erstellen von Pools finden Sie unter „Erstellen eines ZFS-Speicher-Pools“ auf Seite 72.

Alle ZFS-Dateisysteme werden beim Systemstart von ZFS mithilfe des SMF-Dienstes (Service Management Facility) `svc://system/filesystem/local` eingehängt. Dateisysteme werden unter /*path* eingehängt, wobei *path* den Namen des Dateisystems bezeichnet.

Sie können den Standard-Einhängepunkt überschreiben, indem Sie den Befehl `zfs set` verwenden, um die Eigenschaft `mountpoint` auf einen spezifischen Pfad zu setzen. ZFS erstellt den angegebenen Einhängepunkt bei Bedarf automatisch und hängt das entsprechende Dateisystem automatisch ein, wenn der Befehl `zfs mount -a` aufgerufen wird, ohne dass Sie dafür die Datei `/etc/vfstab` ändern müssen.

Die Eigenschaft `mountpoint` wird vererbt. Wenn die Eigenschaft `mountpoint` von `pool/home` beispielsweise auf `/export/stuff` gesetzt ist, erbt `pool/home/user` für seinen Eigenschaftswert `mountpoint` den Wert `/export/stuff/user`.

Um zu verhindern, dass ein Dateisystem eingehängt wird, setzen Sie die Eigenschaft `mountpoint` auf `none`. Außerdem kann mit der Eigenschaft `canmount` bestimmt werden, ob ein Dateisystem eingehängt werden kann. Weitere Informationen zur Eigenschaft `canmount` finden Sie unter „Die Eigenschaft `canmount`" auf Seite 201.

Dateisysteme können auch explizit mithilfe von Legacy-Einhängesystemen verwaltet werden, indem `zfs set` verwendet wird, um die Eigenschaft `mountpoint` auf `legacy` zu setzen. Dadurch wird verhindert, dass ZFS ein Dateisystem automatisch einhängt und verwaltet. Stattdessen müssen Sie Legacy-Dienstprogramme wie die Befehle `mount` und `umount` und die Datei `/etc/vfstab` verwenden. Weitere Informationen zu Legacy-Einhängepunkten finden Sie unter „Legacy-Einhängepunkte" auf Seite 213.

Automatische Einhängepunkte

- Wenn Sie die Eigenschaft `mountpoint` von `legacy` oder `none` auf einen bestimmten Pfad umsetzen, hängt ZFS das betreffende Dateisystem automatisch ein.
- Wenn ZFS ein Dateisystem verwaltet, dieses aber ausgehängt ist und die Eigenschaft `mountpoint` geändert wird, bleibt das Dateisystem weiterhin ausgehängt.

Datasets, deren Eigenschaft `mountpoint` nicht auf `legacy` gesetzt ist, werden von ZFS verwaltet. Im folgenden Beispiel wird ein Dataset erstellt, dessen Einhängepunkt automatisch von ZFS verwaltet wird:

```
# zfs create pool/filesystem
# zfs get mountpoint pool/filesystem
NAME             PROPERTY      VALUE                      SOURCE
pool/filesystem  mountpoint    /pool/filesystem           default
# zfs get mounted pool/filesystem
NAME             PROPERTY      VALUE                      SOURCE
pool/filesystem  mounted       yes                        -
```

Sie können die Eigenschaft `mountpoint` auch explizit setzen (siehe folgendes Beispiel):

```
# zfs set mountpoint=/mnt pool/filesystem
# zfs get mountpoint pool/filesystem
NAME             PROPERTY      VALUE                    SOURCE
pool/filesystem  mountpoint    /mnt                     local
# zfs get mounted pool/filesystem
NAME             PROPERTY      VALUE                    SOURCE
pool/filesystem  mounted       yes                      -
```

Beim Ändern der Eigenschaft `mountpoint` wird das betreffende Dateisystem automatisch aus dem alten Einhängepunkt ausgehängt und in den neuen Einhängepunkt eingehängt. Einhängepunktverzeichnisse werden je nach Bedarf erstellt. Wenn ZFS ein Dateisystem nicht aushängen kann, weil es noch aktiv ist, wird ein Fehler gemeldet, und das Aushängen muss manuell erzwungen werden.

Legacy-Einhängepunkte

Sie können ZFS-Dateisysteme mit Legacy-Dienstprogrammen verwalten, indem Sie die Eigenschaft `mountpoint` auf `legacy` setzen. Legacy-Dateisysteme müssen mithilfe der Befehle `mount` und `umount` sowie der Datei `/etc/vfstab` verwaltet werden. ZFS hängt Legacy-Dateisysteme beim Systemstart nicht automatisch ein, und die ZFS-Befehle `mount` und `umount` funktionieren mit Datasets dieses Typs nicht. Die folgenden Beispiele zeigen die Erstellung und Verwaltung eines ZFS-Datasets im Legacy-Modus:

```
# zfs set mountpoint=legacy tank/home/eschrock
# mount -F zfs tank/home/eschrock /mnt
```

Damit Legacy-Dateisysteme beim Systemstart automatisch eingehängt werden, müssen Sie zur Datei `/etc/vfstab` die entsprechenden Einträge hinzufügen. Das folgende Beispiel zeigt, wie der Eintrag in der Datei `/etc/vfstab` aussehen kann:

```
#device         device        mount           FS       fsck     mount   mount
#to mount       to fsck       point           type     pass     at boot options
#

tank/home/eschrock -        /mnt           zfs         -         yes        -
```

Die Einträge `device to fsck` und `fsck pass` werden auf `-` gesetzt, weil der Befehl `fsck` nicht auf ZFS-Dateisysteme anwendbar ist. Weitere Informationen zur ZFS-Datenintegrität finden Sie unter „Transaktionale Semantik“ auf Seite 47.

Einhängen von ZFS-Dateisystemen

ZFS hängt Dateisysteme beim Erstellen dieser Dateisysteme bzw. beim Systemstart automatisch ein. Der Befehl `zfs mount` muss nur verwendet werden, wenn Einhängeoptionen geändert oder Dateisysteme explizit ein- oder ausgehängt werden müssen.

Beim Aufrufen des Befehls `zfs mount` ohne Argumente werden alle von ZFS verwalteten und gegenwärtig eingehängten Dateisysteme angezeigt. Legacy-Einhängepunkte werden nicht angezeigt. Beispiel:

```
# zfs mount
tank                            /tank
tank/home                       /tank/home
tank/home/bonwick               /tank/home/bonwick
tank/ws                         /tank/ws
```

Mit der Option -a können Sie alle von ZFS verwalteten Dateisysteme einhängen. Legacy-Dateisysteme werden nicht eingehängt. Beispiel:

```
# zfs mount -a
```

Standardmäßig erlaubt ZFS das Einhängen in ein nicht leeres Verzeichnis nicht. Zum Erzwingen des Einhängens in ein nicht leeres Verzeichnis müssen Sie die Option -O verwenden. Beispiel:

```
# zfs mount tank/home/lalt
cannot mount '/export/home/lalt': directory is not empty
use legacy mountpoint to allow this behavior, or use the -O flag
# zfs mount -O tank/home/lalt
```

Legacy-Einhängepunkte müssen mit Legacy-Dienstprogrammen verwaltet werden. Wenn Sie versuchen, dafür ZFS-Befehle zu verwenden, wird ein Fehler ausgegeben. Beispiel:

```
# zfs mount pool/home/billm
cannot mount 'pool/home/billm': legacy mountpoint
use mount(1M) to mount this filesystem
# mount -F zfs tank/home/billm
```

Beim Einhängen eines Dateisystems werden verschiedene Einhängeoptionen verwendet, die auf den zum Dataset gehörenden Eigenschaftswerten beruhen. Zwischen Eigenschaften und Einhängeoptionen besteht der folgende Zusammenhang:

TABELLE 6–4 Eigenschaften von ZFS-Einhängepunkten und Einhängeoptionen

Eigenschaft	Einhängeoption
atime	atime/noatime
devices	devices/nodevices
exec	exec/noexec
nbmand	nbmand/nonbmand
readonly	ro/rw
setuid	setuid/nosetuid
xattr	xattr/noaxttr

Die Einhängeoption nosuid ist ein Alias-Name für nodevices,nosetuid .

Verwenden temporärer Einhängepunkte

Wenn die im vorherigen Abschnitt beschriebenen Optionen explizit mithilfe der Option -o des Befehls zfs mount gesetzt werden, wird der zugehörige Eigenschaftswert temporär überschrieben. Diese Eigenschaftswerte werden vom Befehl zfs get mit dem Wert temporary gemeldet und beim Aushängen des betreffenden Dateisystems auf ihre ursprünglichen Werte zurückgesetzt. Beim Ändern eines Eigenschaftswerts während des Einhängens eines Datasets wird diese Änderung sofort übernommen und überschreibt eventuelle temporäre Einstellungen.

Im folgenden Beispiel wird am Dateisystem tank/home/perrin die Einhängeoption „schreibgeschützt" temporär gesetzt. Es wird angenommen, dass das Dateisystem ausgehängt ist.

```
# zfs mount -o ro tank/home/perrin
```

Zum temporären Ändern eines Eigenschaftswerts eines gegenwärtig eingehängten Dateisystems dient die spezielle Option remount. Im folgenden Beispiel wird die Eigenschaft atime für ein eingehängtes Dateisystem temporär auf off gesetzt.

```
# zfs mount -o remount,noatime tank/home/perrin
# zfs get atime tank/home/perrin
NAME             PROPERTY      VALUE                      SOURCE
tank/home/perrin atime         off                        temporary
```

Weitere Informationen zum Befehl zfs mount finden Sie in der Man Page zfs(1M).

Aushängen von ZFS-Dateisystemen

ZFS-Dateisysteme können mit dem Befehl zfs unmount ausgehängt werden. Der Befehl unmount akzeptiert als Argument entweder den Einhängepunkt oder den Dateisystemnamen.

Im folgenden Beispiel wird ein Dateisystem nach seinem Namen ausgehängt:

```
# zfs unmount tank/home/tabriz
```

Im folgenden Beispiel wird ein Dateisystem nach seinem Einhängepunkt ausgehängt:

```
# zfs unmount /export/home/tabriz
```

Der Befehl unmount schlägt fehl, wenn das betreffende Dateisystem aktiv ist. Ein Aushängen eines Dateisystems kann mit der Option -f erzwungen werden. Gehen Sie beim erzwungenen Aushängen eines Dateisystems äußerst sorgsam vor, wenn der Inhalt dieses Dateisystems noch verwendet wird. da unvorhergesehenes Anwendungsverhalten die Folge davon sein kann.

```
# zfs unmount tank/home/eschrock
cannot unmount '/export/home/eschrock': Device busy
# zfs unmount -f tank/home/eschrock
```

Zum Zweck der Abwärtskompatibilität kann der Legacy-Befehl umount auch zum Aushängen von ZFS-Dateisystemen verwendet werden. Beispiel:

```
# umount /export/home/bob
```

Weitere Informationen zum Befehl zfs unmount finden Sie in der Man Page zfs(1M).

Freigeben und Sperren von ZFS-Dateisystemen

ZFS kann Dateisysteme durch entsprechendes Setzen der Eigenschaft sharenfs automatisch für den Netzwerkzugriff freigeben. Mit dieser Eigenschaft müssen Sie die Datei /etc/dfs/dfstab nicht ändern, wenn ein neues Dateisystem freigegeben wurde. Die Eigenschaft sharenfs ist eine kommagetrennte Liste mit Optionen, die an den Befehl share übergeben wird. Der spezielle Wert on ist ein Aliasname für die Standardoptionen der Netzwerkfreigabe, der allen Benutzern Lese- und Schreibberechtigung gewährt. Der Wert off gibt an, dass das Dateisystem nicht von ZFS verwaltet wird und über herkömmliche Mittel wie z. B. die Datei /etc/dfs/dfstab für den Netzwerkzugriff freigegeben werden kann. Alle Dateisysteme, deren Eigenschaft sharenfs nicht auf off gesetzt ist, werden beim Systemstart freigegeben.

Einstellen der Freigabesemantik

Standardmäßig sind alle Dateisysteme für den Netzwerkzugriff gesperrt. Zum Freigeben eines neuen Dateisystems für den Netzwerkzugriff müssen Sie den Befehl zfs set mit der folgenden Syntax verwenden:

```
# zfs set sharenfs=on tank/home/eschrock
```

Die Eigenschaft sharenfs wird vererbt, und alle Dateisysteme werden bei der Erstellung automatisch für den Netzwerkzugriff freigegeben, wenn deren vererbte Eigenschaft nicht auf off gesetzt ist. Beispiel:

```
# zfs set sharenfs=on tank/home
# zfs create tank/home/bricker
# zfs create tank/home/tabriz
# zfs set sharenfs=ro tank/home/tabriz
```

Die Dateisysteme tank/home/bricker und tank/home/tabriz sind anfänglich mit Schreibzugriff freigegeben, da sie die Eigenschaft sharenfs von tank/home erben. Nach dem Setzen dieser Eigenschaft auf ro (schreibgeschützt) wird das Dateisystem tank/home/tabriz unabhängig davon, welchen Wert die Eigenschaft sharenfs für tank/home hat, schreibgeschützt freigegeben.

Sperren von ZFS-Dateisystemen für den Netzwerkzugriff

Obwohl die meisten Dateisysteme beim Systemstart, der Erstellung und beim Löschen automatisch für den Netzwerkzugriff freigegeben bzw. gesperrt werden, kann es manchmal vorkommen, dass Dateisysteme explizit für den Netzwerkzugriff gesperrt werden müssen. Dazu dient der Befehl `zfs unshare`. Beispiel:

```
# zfs unshare tank/home/tabriz
```

Dieser Befehl sperrt das Dateisystem `tank/home/tabriz` für den Netzwerkzugriff. Zum Sperren aller ZFS-Dateisysteme auf einem System benötigen Sie die Option `-a`.

```
# zfs unshare -a
```

Freigeben von ZFS-Dateisystemen für den Netzwerkzugriff

In den meisten Fällen reicht das automatische ZFS-Verhalten des Freigebens von Dateisystemen beim Systemstart und Erstellen eines Dateisystems für den Normalbetrieb aus. Falls Dateisysteme doch einmal für den Netzwerkzugriff gesperrt werden müssen, können Sie diese mit dem Befehl `zfs share` wieder freigeben. Beispiel:

```
# zfs share tank/home/tabriz
```

Sie können auch alle ZFS-Dateisysteme auf einem System mithilfe der Option `-a` freigeben.

```
# zfs share -a
```

Freigabeverhalten bei Legacy-Dateisystemen

Wenn die Eigenschaft `sharenfs` auf `off` gesetzt ist, versucht ZFS niemals, das betreffende Dateisystem für den Netzwerkzugriff freizugeben bzw. zu sperren. Dieser Wert ermöglicht die Freigabeverwaltung mithilfe herkömmlicher Mittel wie z. B. der Datei `/etc/dfs/dfstab`.

Im Gegensatz zum Befehl `mount` funktionieren die Befehle `share` und `unshare` auch für ZFS-Dateisysteme. Deswegen können Sie ein Dateisystem manuell mit Optionen freigeben, die sich von den Optionen der Eigenschaft `sharenfs` unterscheiden. Davon wird jedoch abgeraten. Sie sollten NFS-Freigaben entweder vollständig von ZFS oder vollständig mithilfe der Datei `/etc/dfs/dfstab` verwalten lassen. Das administrative ZFS-Modell ist einfacher und nicht so aufwändig wie das traditionelle Modell.

Einstellen von ZFS-Kontingenten und -Reservierungen

Mit der Eigenschaft `quota` können Sie die Festplattenkapazität, die ein Dateisystem verwenden kann, beschränken. Darüber hinaus können Sie mit der Eigenschaft `reservation` für ein Dateisystem verfügbare Festplattenkapazität garantieren. Beide Eigenschaften gelten für das Dataset, für das sie gesetzt wurden, und für alle seine untergeordneten Datasets.

Wenn beispielsweise für das Dataset `tank/home` ein Kontingent festgelegt wurde, heißt das, dass die insgesamt von `tank/home` *und allen seinen untergeordneten Datasets* belegte Festplattenkapazität dieses Kontingent nicht überschreiten kann. Genauso wird beim Festlegen einer Reservierung für `tank/home` dieses Dataset *und allen seinen untergeordneten Datasets* dieser Speicherplatz garantiert. Die von einem Dataset und allen seinen untergeordneten Datasets belegte Festplattenkapazität wird von der Eigenschaft `used` verfolgt.

Die Eigenschaften `refquota` und `refreservation` stehen zur Verwaltung von Systemspeicherplatz zur Verfügung. Die von untergeordneten Objekten wie Snapshots und Klonen beanspruchte Festplattenkapazität wird dabei nicht berücksichtigt.

In diesem Solaris-Release können Sie ein *user*- oder *group*-Kontingent für die Festplattenkapazität festlegen, die von Dateien beansprucht wird, die zu einem bestimmten Benutzer oder einer bestimmten Gruppe gehören. Die Kontingenteigenschaften des Benutzers oder der Gruppe können nicht auf einem Volume, einem Dateisystem vor Dateisystem-Version 4 oder einem Pool vor Pool-Version 15 eingerichtet werden.

Beachten Sie die folgenden Faktoren, um festzustellen, welche Kontingent- und Reservierungsfunktionen sich am besten für die Verwaltung Ihrer Dateisysteme anbieten:

- Die Eigenschaften `quota` und `reservation` eignen sich zur Verwaltung von Festplattenkapazität, die durch Datasets und deren untergeordnete Datasets belegt ist.
- Die Eigenschaften `refquota` und `refreservation` eignen sich zur Verwaltung von Festplattenkapazität, die durch Datasets belegt ist.
- Das Setzen der Eigenschaft `refquota` oder `refreservation` auf einen höheren Wert als die Eigenschaft `quota` oder `reservation` hat keine Wirkung. Wenn Sie die Eigenschaft `quota` oder `refquota` setzen, schlagen Vorgänge, bei denen einer dieser Werte überschritten wird, fehl. Es ist möglich, ein Kontingent (`quota`) zu überschreiten, das größer ist als `refquota`. Wenn Snapshot-Blöcke beispielsweise geändert werden, kann tatsächlich eher `quota` als `refquota` überschritten werden.
- Benutzer- und Gruppenkontingente vereinfachen die Verwaltung des Festplattenspeichers bei Vorhandensein vieler Benutzerkonten, zum Beispiel in einer universitären Umgebung.

Weitere Informationen zum Einrichten von Kontingenten und Reservierungen finden Sie unter „Setzen von Kontingenten für ZFS-Dateisysteme" auf Seite 219und „Setzen von Reservierungen für ZFS-Dateisysteme" auf Seite 222.

Setzen von Kontingenten für ZFS-Dateisysteme

Kontingente für ZFS-Dateisysteme können mit den Befehlen `zfs set` und `zfs get` festgelegt und angezeigt werden. Im folgenden Beispiel wird für das Dateisystem `tank/home/bonwick` ein Kontingent von 10 GB gesetzt.

```
# zfs set quota=10G tank/home/bonwick
# zfs get quota tank/home/bonwick
NAME              PROPERTY      VALUE                      SOURCE
tank/home/bonwick quota         10.0G                      local
```

ZFS-Kontingente wirken sich auch auf die Ausgabe der Befehle `list` und `df` aus. Beispiel:

```
# zfs list
NAME                    USED  AVAIL  REFER  MOUNTPOINT
tank/home              16.5K  33.5G  8.50K  /export/home
tank/home/bonwick      15.0K  10.0G  8.50K  /export/home/bonwick
tank/home/bonwick/ws   6.50K  10.0G  8.50K  /export/home/bonwick/ws
# df -h /export/home/bonwick
Filesystem              size   used  avail capacity  Mounted on
tank/home/bonwick        10G     8K    10G     1%    /export/home/bonwick
```

Bitte beachten Sie, dass, obwohl für `tank/home` 33,5 GB Festplattenkapazität verfügbar sind, wegen des für `tank/home/bonwick` gesetzten Kontingents für `tank/home/bonwick` und `tank/home/bonwick/ws` nur 10 GB verfügbar sind.

Kontingente können nicht auf Werte gesetzt werden, die kleiner als der gegenwärtig vom betreffenden Dataset belegte Speicherplatz sind. Beispiel:

```
# zfs set quota=10K tank/home/bonwick
cannot set quota for 'tank/home/bonwick': size is less than current used or
reserved space
```

Es ist möglich, `refquota` für ein Dataset zu setzen, um die durch das Dataset belegbare Festplattenkapazität einzuschränken. Dieser absolute Grenzwert berücksichtigt keine Festplattenkapazität, die von untergeordneten Objekten belegt wird. Beispiel:

```
# zfs set refquota=10g students/studentA
# zfs list
NAME                 USED  AVAIL  REFER  MOUNTPOINT
profs                106K  33.2G    18K  /profs
students            57.7M  33.2G    19K  /students
students/studentA   57.5M  9.94G  57.5M  /students/studentA
# zfs snapshot students/studentA@today
# zfs list
NAME                       USED  AVAIL  REFER  MOUNTPOINT
profs                      106K  33.2G    18K  /profs
students                  57.7M  33.2G    19K  /students
students/studentA         57.5M  9.94G  57.5M  /students/studentA
students/studentA@today       0      -  57.5M  -
```

Sie können ein weiteres Kontingent für ein Dataset festlegen, um die Verwaltung der durch Snapshots belegten Festplattenkapazität zu erleichtern. Beispiel:

```
# zfs set quota=20g students/studentA
# zfs list
NAME                       USED  AVAIL  REFER  MOUNTPOINT
profs                      106K  33.2G    18K  /profs
students                  57.7M  33.2G    19K  /students
students/studentA         57.5M  9.94G  57.5M  /students/studentA
students/studentA@today       0      -  57.5M  -
```

In diesem Szenario kann `studentA` den mit „`refquota`" festgelegten absoluten Grenzwert (10 GB) erreichen und selbst bei vorhandenen Snapshots wiederherzustellende Dateien entfernen.

Im obigen Beispiel wird das kleinere der beiden Kontingente (10 GB im Vergleich mit 20 GB) in der Ausgabe von `zfs list` angezeigt. Zum Anzeigen der Werte beider Kontingente verwenden Sie den Befehl `zfs get`. Beispiel:

```
# zfs get refquota,quota students/studentA
NAME               PROPERTY  VALUE              SOURCE
students/studentA  refquota  10G                local
students/studentA  quota     20G                local
```

Einrichten von Benutzer- und Gruppenkontingenten auf einem ZFS-Dateisystem

Unter Verwendung des Befehls `zfs userquota` oder `zfs groupquota` können Sie ein Benutzer- oder Gruppenkontingent wie folgt einrichten. Beispiel:

```
# zfs create students/compsci
# zfs set userquota@student1=10G students/compsci
# zfs create students/labstaff
# zfs set groupquota@staff=20GB students/labstaff
```

Zeigen Sie das aktuelle Benutzer- oder Gruppenkontingent wie folgt an:

```
# zfs get userquota@student1 students/compsci
NAME              PROPERTY            VALUE               SOURCE
students/compsci  userquota@student1  10G                 local
# zfs get groupquota@staff students/labstaff
NAME               PROPERTY          VALUE             SOURCE
students/labstaff  groupquota@staff  20G               local
```

Sie können die von allgemeinen Benutzern und Gruppen belegte Festplattenkapazität durch Abfrage der folgenden Eigenschaften anzeigen:

```
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
POSIX User  root      227M   none
POSIX User  student1  455M    10G
# zfs groupspace students/labstaff
TYPE         NAME   USED  QUOTA
POSIX Group  root   217M   none
POSIX Group  staff  217M    20G
```

Um die von einzelnen Benutzern oder Gruppen belegte Festplattenkapazität zu ermitteln, fragen Sie die folgenden Eigenschaften ab:

```
# zfs get userused@student1 students/compsci
NAME              PROPERTY          VALUE             SOURCE
students/compsci  userused@student1  455M             local
# zfs get groupused@staff students/labstaff
NAME               PROPERTY        VALUE            SOURCE
students/labstaff  groupused@staff  217M            local
```

Die Eigenschaften der Benutzer- und Gruppenkontingente werden nicht über den Befehl `zfs get all` *dataset* angezeigt. Dieser führt die Eigenschaften aller anderen Dateisysteme auf.

Sie können ein Benutzer- oder Gruppenkontingent folgendermaßen entfernen:

```
# zfs set userquota@user1=none students/compsci
# zfs set groupquota@staff=none students/labstaff
```

ZFS-Benutzer- und Gruppenkontingente in ZFS-Dateisystemen besitzen folgende Merkmale:

- Ein in einem übergeordneten Dateisystem eingerichtetes Benutzer- oder Gruppenkontingent wird nicht automatisch von einem untergeordneten Dateisystem übernommen.
- Allerdings wird das Benutzer- oder Gruppenkontingent angewendet, wenn ein Klon oder Snapshot aus einem Dateisystem mit einem Benutzer- oder Gruppenkontingent erstellt wird. Ebenso wird ein Benutzer- oder Gruppenkontingent in das Dateisystem aufgenommen, wenn mit dem Befehl `zfs send` ein Datenstrom erstellt wird, auch ohne Anwendung der Option `-R`.
- Benutzer ohne Privilegien haben nur Zugriff auf ihre eigene Festplattenkapazität. Der Root-Benutzer oder ein Benutzer, dem das Privileg `userused` oder `groupused` erteilt wurde, kann auf Informationen zur Berechnung der Festplattenkapazität sämtlicher Benutzer und Gruppen zugreifen.
- Die Eigenschaften `userquota` und `groupquota` können nicht auf ZFS-Volumes, auf einem Dateisystem vor Dateisystem-Version 4 oder auf einem Pool vor Pool-Version 15 eingerichtet werden.

Das Inkrafttreten von Benutzer- und Gruppenkontingenten kann einige Sekunden dauern. Die Verzögerung kann bedeuten, dass Benutzer ihr Kontingent überschreiten, bevor das System dies registriert und weitere Schreibvorgänge mit der Fehlermeldung `EDQUOT` zurückweist.

Mit dem Legacy-Befehl `quota` können Sie Benutzerkontingente in einer NFS-Umgebung überprüfen, zum Beispiel beim Einhängen eines ZFS-Dateisystems. Ohne Auswahl von Optionen zeigt der Befehl `quota` nur Ergebnisse, wenn das Benutzerkontingent überschritten wird. Beispiel:

```
# zfs set userquota@student1=10m students/compsci
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
```

```
POSIX User  root      227M   none
POSIX User  student1  455M    10M
# quota student1
Block limit reached on /students/compsci
```

Wenn Sie das Benutzerkontingent zurücksetzen und die Begrenzung des Kontingents nicht länger überschritten ist, können Sie zur Überprüfung des Benutzerkontingents den Befehl `quota -v` verwenden. Beispiel:

```
# zfs set userquota@student1=10GB students/compsci
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
POSIX User  root      227M   none
POSIX User  student1  455M    10G
# quota student1
# quota -v student1
Disk quotas for student1 (uid 201):
Filesystem     usage  quota  limit    timeleft  files  quota  limit    timeleft
/students/compsci
              466029 10485760 10485760
```

Setzen von Reservierungen für ZFS-Dateisysteme

Unter einer ZFS-*Reservierung* versteht man eine Zuweisung von Festplattenkapazität aus dem Pool, die einem Dataset garantiert ist. Sie können keine Reservierungen vornehmen, wenn die angeforderte Festplattenkapazität im Pool nicht zur Verfügung steht. Der Gesamtbetrag aller noch ausstehenden und nicht belegten Reservierungen darf die Gesamtsumme der nicht belegten Festplattenkapazität im Pool nicht überschreiten. ZFS-Reservierungen können mit den Befehlen `zfs set` und `zfs get` festgelegt und angezeigt werden. Beispiel:

```
# zfs set reservation=5G tank/home/moore
# zfs get reservation tank/home/moore
NAME             PROPERTY     VALUE   SOURCE
tank/home/moore  reservation  5G      local
```

ZFS-Reservierungen können sich auf die Ausgabe des Befehls `zfs list` auswirken. Beispiel:

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank/home             5.00G  33.5G  8.50K  /export/home
tank/home/moore       15.0K  33.5G  8.50K  /export/home/moore
```

Bitte beachten Sie, dass `tank/home` 5 GB Festplattenkapazität belegt, obwohl `tank/home` und seinen untergeordneten Dateisysteme tatsächlich viel weniger als 5 GB Festplattenkapazität belegen. Der belegte Speicherplatz berücksichtigt die Reservierung für `tank/home/moore`. Reservierungen werden in die belegten Festplattenkapazität des übergeordneten Datasets einbezogen und auf sein Kontingent bzw. seine Reservierung (bzw. beide) angerechnet.

```
# zfs set quota=5G pool/filesystem
# zfs set reservation=10G pool/filesystem/user1
cannot set reservation for ’pool/filesystem/user1’: size is greater than
available space
```

Ein Dataset kann mehr Festplattenkapazität belegen, als für seine Reservierung vorgesehen ist, solange nicht reservierter Speicherplatz im Pool vorhanden ist und der aktuell vom Dataset belegte Speicherplatz unter seinem Kontingent liegt. Ein Dataset kann keine Festplattenkapazität belegen, die für ein anderes Dataset reserviert wurde.

Reservierungen sind nicht kumulativ. Das bedeutet, dass durch einen zweiten Aufruf von `zfs set` zum Setzen einer Reservierung der Speicherplatz der zweiten Reservierung nicht zum Speicherplatz der vorhandenen Reservierung addiert wird. Die zweite Reservierung ersetzt stattdessen die erste Reservierung. Beispiel:

```
# zfs set reservation=10G tank/home/moore
# zfs set reservation=5G tank/home/moore
# zfs get reservation tank/home/moore
NAME             PROPERTY      VALUE                        SOURCE
tank/home/moore  reservation   5.00G                        local
```

Durch Setzen der Reservierung `refreservation` können Sie einem Dataset Festplattenkapazität garantieren, in der die von Snapshots und Klonen belegte Festplattenkapazität nicht berücksichtigt ist. Diese Reservierung wird in die Berechnung der Speicherplatzkapazität für das diesem Dataset übergeordnete Dataset einbezogen und auf die Kontingente und Reservierung für das übergeordnete Dataset angerechnet. Beispiel:

```
# zfs set refreservation=10g profs/prof1
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
profs                      10.0G  23.2G    19K  /profs
profs/prof1                  10G  33.2G    18K  /profs/prof1
```

Sie können außerdem für dasselbe Dataset eine Reservierung festlegen, um Speicherplatz für das Dataset und für Snapshots zu garantieren. Beispiel:

```
# zfs set reservation=20g profs/prof1
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
profs                      20.0G  13.2G    19K  /profs
profs/prof1                  10G  33.2G    18K  /profs/prof1
```

Normale Reservierungen werden in die Berechnung des belegten Speicherplatzes durch das übergeordnete Dataset einbezogen.

Im obigen Beispiel wird das kleinere der beiden Kontingente (10 GB im Vergleich mit 20 GB) in der Ausgabe von `zfs list` angezeigt. Zum Anzeigen der Werte beider Kontingente verwenden Sie den Befehl `zfs get`. Beispiel:

```
# zfs get reservation,refreserv profs/prof1
NAME         PROPERTY        VALUE        SOURCE
profs/prof1  reservation     20G          local
profs/prof1  refreservation  10G          local
```

Wenn `refreservation` gesetzt ist, wird ein Snapshot nur zugelassen, wenn außerhalb dieser Reservierung genügend nicht reservierter Speicherplatz im Pool vorhanden ist, um die Menge der aktuell *referenzierten* Byte im Dataset aufzunehmen.



Arbeiten mit Oracle Solaris ZFS-Snapshots und -Klonen

Dieses Kapitel enthält Informationen zum Erstellen und Verwalten von Oracle Solaris ZFS-Snapshots und -Klonen. Außerdem finden Sie hier auch Informationen zum Speichern von Snapshots.

Dieses Kapitel enthält folgende Abschnitte:

Überblick über ZFS-Snapshots

Ein *Snapshot* ist eine schreibgeschützte Kopie eines Dateisystems bzw. Volumes. Snapshots können sehr schnell erstellt werden und belegen anfänglich keine zusätzliche Festplattenkapazität im Pool. Mit der Änderung von Daten innerhalb des aktiven Datasets belegt der Snapshot jedoch schrittweise mehr Festplattenkapazität, da er Verweise auf die älteren Daten speichert und so ein Löschen dieser Daten verhindert

ZFS-Snapshots besitzen die folgenden Leistungsmerkmale:

- Sie bleiben auch nach Systemneustarts wirksam.
- Die theoretische Maximalanzahl möglicher Snapshots beträgt 2^{64}.
- Snapshots verwenden keine separaten Zusatzspeicher. Snapshots belegen Festplattenkapazität direkt im gleichen Speicher-Pool, wie auch das Dateisystem oder Volume, aus dem sie erstellt wurden.

- Rekursive Snapshots werden schnell in einem unteilbaren Vorgang erstellt. Snapshots werden entweder zusammen (d. h. alle auf einmal) oder gar nicht erstellt. Der Vorteil solcher unteilbarer Snapshots besteht darin, dass die Snapshot-Daten auch bei Dateisystemhierarchien zu einem einzigen konsistenten Zeitpunkt erstellt werden.

Auf Snapshots von Volumes kann nicht direkt zugegriffen werden, aber sie können geklont, als Sicherungskopie gesichert und wiederhergestellt werden. Informationen zum Erstellen von Sicherungskopien für ZFS-Snapshots finden Sie unter „Senden und Empfangen von ZFS-Daten“ auf Seite 234.

- „Erstellen und Löschen von ZFS-Snapshots“ auf Seite 226
- „Anzeigen von und Zugreifen auf ZFS-Snapshots“ auf Seite 229
- „Wiederherstellen eines früheren ZFS-Snapshots“ auf Seite 231

Erstellen und Löschen von ZFS-Snapshots

Snapshots werden mithilfe des Befehls `zfs snapshot` erstellt. Diesem wird als einziges Argument der Name des zu erstellenden Snapshots übergeben. Der Snapshot-Name ist wie folgt anzugeben:

filesystem@snapname
volume@snapname

Der Snapshot-Name muss den unter „Konventionen für das Benennen von ZFS-Komponenten“ auf Seite 51 aufgeführten Benennungskonventionen genügen.

Im folgenden Beispiel wird ein Snapshot des Dateisystems `tank/home/ahrens` mit dem Namen `friday` erstellt.

```
# zfs snapshot tank/home/ahrens@friday
```

Mithilfe der Option `-r` können Sie Snapshots für alle untergeordneten Dateisysteme erstellen. Beispiel:

```
# zfs snapshot -r tank/home@now
# zfs list -t snapshot
NAME                       USED  AVAIL  REFER  MOUNTPOINT
rpool/ROOT/zfs2BE@zfs2BE  78.3M      -  4.53G  -
tank/home@now                 0      -    26K  -
tank/home/ahrens@now          0      -   259M  -
tank/home/anne@now            0      -   156M  -
tank/home/bob@now             0      -   156M  -
tank/home/cindys@now          0      -   104M  -
```

Snapshots besitzen keine konfigurierbaren Eigenschaften, und es können für Snapshots auch keine Dataset-Eigenschaften gesetzt werden. Beispiel:

```
# zfs set compression=on tank/home/ahrens@now
cannot set compression property for 'tank/home/ahrens@now': snapshot
properties cannot be modified
```

Snapshots werden mithilfe des Befehls `zfs destroy` gelöscht. Beispiel:

```
# zfs destroy tank/home/ahrens@now
```

Ein Dataset kann nicht gelöscht werden, wenn Snapshots dieses Datasets vorhanden sind. Beispiel:

```
# zfs destroy tank/home/ahrens
cannot destroy 'tank/home/ahrens': filesystem has children
use '-r' to destroy the following datasets:
tank/home/ahrens@tuesday
tank/home/ahrens@wednesday
tank/home/ahrens@thursday
```

Wenn von einem Snapshot Klone erstellt wurden, müssen diese zuerst gelöscht werden, bevor das Löschen des Snapshots möglich ist.

Weitere Informationen zum Unterbefehl `destroy` finden Sie unter „Löschen eines ZFS-Dateisystems“ auf Seite 187.

Aufbewahren von ZFS-Snapshots

Wenn Sie verschiedene automatische Snapshot-Richtlinien verwenden und dadurch ältere Snapshots durch `zfs receive` gelöscht werden, weil sie nicht mehr auf der Sendeseite vorhanden sind, können Sie die Snapshot-Aufbewahrungsfunktion verwenden.

Durch die *Aufbewahrung* eines Snapshots wird verhindert, dass er gelöscht wird. Außerdem ermöglicht diese Funktion das Löschen eines Snapshots zusammen mit Klonen in Abhängigkeit von der Entfernung des letzten Klons mithilfe des Befehls `zfs destroy -d`. Jeder Snapshot besitzt eine zugeordnete Benutzerreferenzzählung, die bei Null beginnt. Dieser Wert wird erhöht, wenn ein weiterer Snapshot aufbewahrt wird, und verringert, wenn eine Aufbewahrung beendet wird.

In der Vorgängerversion von Solaris konnte ein Snapshot nur dann mithilfe des Befehls `zfs destroy` gelöscht werden, wenn er keine Klone hatte. In dieser Solaris-Version muss zudem der Wert der Benutzerreferenzzählung des Snapshots auf Null stehen.

Sie können einen Snapshot oder eine Gruppe von Snapshots aufbewahren. Anhand der folgenden Syntax wird beispielsweise ein Aufbewahrungs-Tag, `keep`, für `tank/home/cindys/snap@1` gesetzt.

```
# zfs hold keep tank/home/cindys@snap1
```

Sie können die Option `-r` verwenden, um die Snapshots aller untergeordneten Dateisystem rekursiv aufzubewahren. Beispiel:

```
# zfs snapshot -r tank/home@now
# zfs hold -r keep tank/home@now
```

Mithilfe dieser Syntax wird eine einzelne Referenz, `keep`, zu einem Snapshot oder einer Gruppe von Snapshots hinzugefügt. Jeder Snapshot besitzt einen eigenen Tag-Namensraum. Die Aufbewahrungs-Tags innerhalb dieses Namensraums müssen eindeutig sein. Wenn ein Snapshot aufbewahrt wird, kann er nicht mithilfe des Befehls `zfs destroy` gelöscht werden. Beispiel:

```
# zfs destroy tank/home/cindys@snap1
cannot destroy 'tank/home/cindys@snap1': dataset is busy
```

Wenn Sie einen aufbewahrten Snapshot löschen möchten, verwenden Sie die Option `-d`. Beispiel:

```
# zfs destroy -d tank/home/cindys@snap1
```

Verwenden Sie den Befehl `zfs holds`, um eine Liste der aufbewahrten Snapshots anzuzeigen. Beispiel:

```
# zfs holds tank/home@now
NAME           TAG   TIMESTAMP
tank/home@now  keep  Thu Jul 15 11:25:39 2010

# zfs holds -r tank/home@now
NAME                  TAG   TIMESTAMP
tank/home/cindys@now  keep  Thu Jul 15 11:25:39 2010
tank/home/mark@now    keep  Thu Jul 15 11:25:39 2010
tank/home@now         keep  Thu Jul 15 11:25:39 2010
```

Sie können den Befehl `zfs release` verwenden, um einen aufbewahrten Snapshot oder eine Gruppe aufbewahrter Snapshots freizugeben. Beispiel:

```
# zfs release -r keep tank/home@now
```

Ist ein Snapshot freigegeben, kann er mithilfe des Befehls `zfs destroy` gelöscht werden. Beispiel:

```
# zfs destroy -r tank/home@now
```

Zwei neue Eigenschaften liefern Informationen zur Aufbewahrung von Snapshots:

- Die Eigenschaft `defer_destroy` ist aktiviert (`on`), wenn der Snapshot zur späteren Löschung mithilfe des Befehls `zfs destroy -d` vorgesehen ist. Anderenfalls ist die Eigenschaft deaktiviert (`off`).
- Die Eigenschaft `userrefs` dient zur Angabe der Anzahl der Aufbewahrungen des Snapshots und wird auch als Benutzerreferenzzählung bezeichnet.

Umbenennen von ZFS-Snapshots

Sie können Snapshots umbenennen. Allerdings müssen Snapshots innerhalb des Pools und Datasets, in dem sie erstellt wurden, umbenannt werden. Beispiel:

```
# zfs rename tank/home/cindys@083006 tank/home/cindys@today
```

Außerdem entspricht die folgende Kurzsyntax der obigen Syntax:

```
# zfs rename tank/home/cindys@083006 today
```

Der folgende Vorgang zum Umbenennen eines Snapshots wird nicht unterstützt, da sich Ziel-Pool und -Dateisystem von dem Pool und Dateisystem unterscheiden, in denen der betreffende Snapshot erstellt wurde:

```
# zfs rename tank/home/cindys@today pool/home/cindys@saturday
cannot rename to 'pool/home/cindys@today': snapshots must be part of same
dataset
```

Sie können Snapshots mithilfe des Befehls `zfs rename - r` rekursiv umbenennen. Beispiel:

```
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
users                       270K  16.5G    22K  /users
users/home                   76K  16.5G    22K  /users/home
users/home@yesterday           0      -    22K  -
users/home/markm             18K  16.5G    18K  /users/home/markm
users/home/markm@yesterday     0      -    18K  -
users/home/marks             18K  16.5G    18K  /users/home/marks
users/home/marks@yesterday     0      -    18K  -
users/home/neil              18K  16.5G    18K  /users/home/neil
users/home/neil@yesterday      0      -    18K  -
# zfs rename -r users/home@yesterday @2daysago
# zfs list -r users/home
NAME                        USED  AVAIL  REFER  MOUNTPOINT
users/home                   76K  16.5G    22K  /users/home
users/home@2daysago            0      -    22K  -
users/home/markm             18K  16.5G    18K  /users/home/markm
users/home/markm@2daysago      0      -    18K  -
users/home/marks             18K  16.5G    18K  /users/home/marks
users/home/marks@2daysago      0      -    18K  -
users/home/neil              18K  16.5G    18K  /users/home/neil
users/home/neil@2daysago       0      -    18K  -
```

Anzeigen von und Zugreifen auf ZFS-Snapshots

Sie können das Anzeigen von Snapshot-Listen in der `zfs list`-Ausgabe durch Verwenden der Pool-Eigenschaft `listsnapshots` aktivieren oder deaktivieren. Diese Eigenschaft ist standardmäßig aktiviert.

Wenn Sie diese Eigenschaft deaktivieren, können Sie Snapshot-Informationen mit dem Befehl `zfs list -t snapshot` anzeigen. Oder aktivieren Sie die Pool-Eigenschaft `listsnapshots`. Beispiel:

```
# zpool get listsnapshots tank
NAME  PROPERTY       VALUE      SOURCE
tank  listsnapshots  on        default
```

```
# zpool set listsnapshots=off tank
# zpool get listsnapshots tank
NAME  PROPERTY       VALUE      SOURCE
tank  listsnapshots  off         local
```

Snapshots befinden sich im Verzeichnis `.zfs/snapshot` des Stammverzeichnisses des Dateisystems. Wenn das Dateisystem `tank/home/ahrens` beispielsweise in `/home/ahrens` eingehängt ist, befinden sich die Daten des Snapshots `tank/home/ahrens@thursday` im Verzeichnis `/home/ahrens/.zfs/snapshot/thursday` .

```
# ls /tank/home/ahrens/.zfs/snapshot
tuesday wednesday thursday
```

Snapshots können wie folgt angezeigt werden:

```
# zfs list -t snapshot
NAME                         USED  AVAIL  REFER  MOUNTPOINT
pool/home/anne@monday           0      -   780K  -
pool/home/bob@monday            0      -  1.01M  -
tank/home/ahrens@tuesday    8.50K      -   780K  -
tank/home/ahrens@wednesday 8.50K      -  1.01M  -
tank/home/ahrens@thursday       0      -  1.77M  -
tank/home/cindys@today      8.50K      -   524K  -
```

Snapshots, die für ein bestimmtes Dateisystem erstellt wurden, können wie folgt angezeigt werden:

```
# zfs list -r -t snapshot -o name,creation tank/home
NAME                   CREATION
tank/home@now          Wed Jun 30 16:16 2010
tank/home/ahrens@now  Wed Jun 30 16:16 2010
tank/home/anne@now    Wed Jun 30 16:16 2010
tank/home/bob@now     Wed Jun 30 16:16 2010
tank/home/cindys@now  Wed Jun 30 16:16 2010
```

Berechnung von Festplattenkapazität für ZFS-Snapshots

Nach der Erstellung eines Snapshots wird dessen Festplattenkapazität anfänglich vom Snapshot und vom Dateisystem (sowie eventuell von früheren Snapshots) gemeinsam genutzt. Wenn sich ein Dateisystem mit der Zeit ändert, wird gemeinsam genutzte Festplattenkapazität dann nur noch vom Snapshot belegt und in die Berechnung des vom Snapshot belegten Speicherplatzes (Eigenschaft `used`) einbezogen. Darüber hinaus kann durch das Löschen von Snapshots die Festplattenkapazität, die Snapshots eindeutig zugewiesen ist (und deswegen in der Eigenschaft *used* angegeben ist und somit belegt wird), größer werden.

Der Eigenschaftswert `referenced` des Speicherlatzes eines Snapshots entspricht dem des Dateisystems zum Zeitpunkt der Erstellung dieses Snapshots.

Sie können zusätzliche Informationen darüber erhalten, wie die Werte der Eigenschaft `used` belegt werden. Neue schreibgeschützte Dateisystem-Eigenschaften beschreiben die Belegung von Festplattenkapazität für Klone, Dateisysteme und Volumes. Beispiel:

```
$ zfs list -o space
# zfs list -ro space tank/home
NAME                  AVAIL   USED  USEDSNAP  USEDDS  USEDREFRESERV  USEDCHILD
tank/home             66.3G   675M         0     26K              0       675M
tank/home@now             -      0         -       -              -          -
tank/home/ahrens      66.3G   259M         0    259M              0          0
tank/home/ahrens@now      -      0         -       -              -          -
tank/home/anne        66.3G   156M         0    156M              0          0
tank/home/anne@now        -      0         -       -              -          -
tank/home/bob         66.3G   156M         0    156M              0          0
tank/home/bob@now         -      0         -       -              -          -
tank/home/cindys      66.3G   104M         0    104M              0          0
tank/home/cindys@now      -      0         -       -              -          -
```

Eine Beschreibung dieser Eigenschaften können Sie Tabelle 6–1 entnehmen.

Wiederherstellen eines früheren ZFS-Snapshots

Mit dem Befehl `zfs rollback` können Sie alle Änderungen rückgängig machen, die seit der Erstellung eines bestimmten Snapshots an einem Dateisystem vorgenommen wurden. Im Dateisystem wird der Zustand zum Zeitpunkt der Erstellung des betreffenden Snapshots wiederhergestellt. Standardmäßig stellt dieser Befehl stets den Zustand des zuletzt gemachten Snapshots wieder her.

Damit das Dateisystem im Zustand eines früheren Snapshots wiederhergestellt werden kann, müssen alle dazwischen liegenden Snapshots gelöscht werden. Frühere Snapshots können mithilfe der Option `-r` gelöscht werden.

Wenn Klone dazwischen liegender Snapshots vorhanden sind, müssen auch diese Klone mithilfe der Option `-R` gelöscht werden.

Hinweis – Das Dateisystem, dessen früherer Zustand wiederhergestellt werden soll, wird aus- und wieder eingehängt, wenn es gerade eingehängt ist. Wenn das betreffende Dateisystem nicht ausgehängt werden kann, schlägt die Wiederherstellung des früheren Zustands fehl. Die Option `-f` erzwingt bei Bedarf das Aushängen des Dateisystems.

Im folgenden Beispiel wird das Dateisystem `tank/home/ahrens` auf den Snapshot mit dem Namen `tuesday` zurückgesetzt:

```
# zfs rollback tank/home/ahrens@tuesday
cannot rollback to 'tank/home/ahrens@tuesday': more recent snapshots exist
use '-r' to force deletion of the following snapshots:
tank/home/ahrens@wednesday
tank/home/ahrens@thursday
# zfs rollback -r tank/home/ahrens@tuesday
```

In diesem Beispiel wurden die Snapshots `wednesday` und `thursday` gelöscht, da Sie den Zustand des davor liegenden Snapshots `tuesday` wiederhergestellt haben.

```
# zfs list -r -t snapshot -o name,creation tank/home/ahrens
NAME                   CREATION
tank/home/ahrens@now  Wed Jun 30 16:16 2010
```

Überblick über ZFS-Klone

Ein *Klon* ist ein schreibbares Volume bzw. Dateisystem, dessen anfänglicher Inhalt gleich dem des Datasets ist, von dem er erstellt wurde. Ebenso wie Snapshots werden auch Klone sehr schnell erstellt und belegen anfänglich keine zusätzliche Festplattenkapazität. Darüber hinaus können Sie Snapshots von Klonen erstellen.

Klone können nur von Snapshots erstellt werden. Beim Klonen eines Snapshots wird zwischen dem Klon und dem Snapshot eine implizite Abhängigkeit erstellt. Obwohl der betreffende Klon an anderer Stelle der Dataset-Hierarchie erstellt wird, kann der ursprüngliche Snapshot nicht gelöscht werden, solange von ihm ein Klon vorhanden ist. Die Eigenschaft `origin` enthält diese Abhängigkeit, und mit dem Befehl `zfs destroy` können solche Abhängigkeiten aufgelistet werden, falls sie vorhanden sind.

Klone erben keine Eigenschaften von dem Dataset, von dem sie erstellt wurden. Mit den Befehlen `zfs get` und `zfs set` können Sie die Eigenschaften eines geklonten Datasets anzeigen und ändern. Weitere Informationen zum Setzen von Eigenschaften von ZFS-Datasets finden Sie unter „Setzen von ZFS-Eigenschaften“ auf Seite 206.

Da ein Klon seine gesamte Festplattenkapazität anfänglich mit dem ursprünglichen Snapshot gemeinsam nutzt, ist sein Eigenschaftswert `used` zu Beginn auf null gesetzt. Wenn am Klon Änderungen vorgenommen werden, belegt er dementsprechend mehr Festplattenkapazität. Die Eigenschaft `used` des ursprünglichen Snapshots berücksichtigt keinen vom Klon belegten Speicherplatz.

- „Erstellen eines ZFS-Klons“ auf Seite 232
- „Löschen eines ZFS-Klons“ auf Seite 233
- „Ersetzen eines ZFS-Dateisystems durch einen ZFS-Klon“ auf Seite 233

Erstellen eines ZFS-Klons

Klone werden mit dem Befehl `zfs clone` erstellt. Sie müssen den Snapshot, von dem der Klon erstellt werden soll, sowie den Namen des neuen Dateisystems bzw. Volumes angeben. Das neue Dateisystem bzw. Volume kann sich an beliebiger Stelle innerhalb der ZFS-Hierarchie befinden. Der Typ (z. B. Dateisystem oder Volume) des neuen Datasets entspricht dem des Snapshots, aus dem der Klon erstellt wurde. Sie können von einem Dateisystem in einem anderen Pool als dem des Snapshots des ursprünglichen Dateisystems keinen Klon erstellen.

Im folgenden Beispiel wird ein neuer Klon `tank/home/ahrens/bug123` mit dem gleichen anfänglichen Inhalt wie der des Snapshots `tank/ws/gate@gestern` erstellt:

```
# zfs snapshot tank/ws/gate@yesterday
# zfs clone tank/ws/gate@yesterday tank/home/ahrens/bug123
```

Im folgenden Beispiel wird für einen temporären Benutzer aus dem Snapshot `projects/newproject@today` ein geklonter Arbeitsbereich namens `projects/teamA/tempuser` erstellt. Danach werden die Eigenschaften des geklonten Arbeitsbereichs gesetzt.

```
# zfs snapshot projects/newproject@today
# zfs clone projects/newproject@today projects/teamA/tempuser
# zfs set sharenfs=on projects/teamA/tempuser
# zfs set quota=5G projects/teamA/tempuser
```

Löschen eines ZFS-Klons

ZFS-Klone werden mithilfe des Befehls `zfs destroy` gelöscht. Beispiel:

```
# zfs destroy tank/home/ahrens/bug123
```

Bevor ein übergeordneter Snapshot gelöscht werden kann, müssen zunächst seine Klone gelöscht werden.

Ersetzen eines ZFS-Dateisystems durch einen ZFS-Klon

Mit dem Befehl `zfs promote` können Sie ein aktives ZFS-Dateisystem durch einen Klon dieses Dateisystems ersetzen. Diese Funktion ermöglicht das Klonen und Ersetzen von Dateisystemen, sodass das *ursprüngliche* Dateisystem der Klon des betreffenden Dateisystems werden kann. Darüber hinaus ermöglicht diese Funktion das Löschen des Dateisystems, von dem der Klon ursprünglich erstellt wurde. Ohne diese ?Klon-Promotion“ können ursprüngliche Dateisysteme, in denen aktive Klone enthalten sind, nicht gelöscht werden. Weitere Informationen zum Löschen von Klonen finden Sie unter „Löschen eines ZFS-Klons“ auf Seite 233.

Im folgenden Beispiel wird das Dateisystem `tank/test/produktA` geklont. Anschließend wird das geklonte Dateisystem (`tank/test/produktAbeta`) zum ursprünglichen Dateisystem `tank/test/produktA` gemacht.

```
# zfs create tank/test
# zfs create tank/test/productA
# zfs snapshot tank/test/productA@today
# zfs clone tank/test/productA@today tank/test/productAbeta
# zfs list -r tank/test
NAME                      USED  AVAIL  REFER  MOUNTPOINT
tank/test                 104M  66.2G    23K  /tank/test
tank/test/productA        104M  66.2G   104M  /tank/test/productA
```

```
tank/test/productA@today      0      -   104M  -
tank/test/productAbeta        0  66.2G   104M  /tank/test/productAbeta
# zfs promote tank/test/productAbeta
# zfs list -r tank/test
NAME                         USED  AVAIL  REFER  MOUNTPOINT
tank/test                    104M  66.2G    24K  /tank/test
tank/test/productA              0  66.2G   104M  /tank/test/productA
tank/test/productAbeta       104M  66.2G   104M  /tank/test/productAbeta
tank/test/productAbeta@today    0      -   104M  -
```

In dieser Ausgabe des Befehls `zfs list` sehen Sie, dass die Festplattenkapazitätsangabe des ursprünglichen Dateisystems `produktA` durch die des Dateisystems `produktAbeta` ersetzt wurde.

Sie können den Ersetzungsvorgang durch Umbenennen der Dateisysteme abschließen. Beispiel:

```
# zfs rename tank/test/productA tank/test/productAlegacy
# zfs rename tank/test/productAbeta tank/test/productA
# zfs list -r tank/test
```

Optional können Sie auch das alte Dateisystem entfernen. Beispiel:

```
# zfs destroy tank/test/productAlegacy
```

Senden und Empfangen von ZFS-Daten

Der Befehl `zfs send` erstellt von Snapshots Datenstrominstanzen, die auf die Standardausgabe geschrieben werden. Standardmäßig wird ein vollständiger Datenstrom erzeugt. Sie können die Ausgabe in eine Datei oder ein anderes Dateisystem umleiten. Der Befehl `zfs receive` erstellt einen Snapshot, dessen Inhalt in einem Datenstrom auf der Standardeingabe angegeben wird. Wenn ein vollständiger Datenstrom gelesen wurde, wird ein neues Dateisystem erstellt. Mit diesen Befehlen können Sie ZFS-Snapshot-Daten senden und ZFS-Snapshot-Daten und Dateisysteme empfangen. Siehe hierzu die Beispiele im nachfolgenden Abschnitt.

- „Senden von ZFS-Snapshots“ auf Seite 236
- „Empfangen von ZFS-Snapshots“ auf Seite 237
- „Senden und Empfangen komplexer ZFS-Snapshot-Datenströme“ auf Seite 238
- „Sichern von ZFS-Daten mit anderen Softwarepaketen zur Erstellung von Sicherungskopien“ auf Seite 235

Die folgenden Lösungen zum Sichern von ZFS-Daten stehen zur Verfügung:

- **Sicherungsprodukte für Unternehmen** – Wenn Sie Bedarf an folgenden Leistungsmerkmalen haben, sollten Sie eine Unternehmenslösung für die Datensicherung in Betracht ziehen:
  - Wiederherstellung auf Dateibasis
  - Überprüfung der Datensicherungsmedien

  - Medienverwaltung
- **Dateisystem-Snapshots und Wiederherstellen des früheren Zustands eines Dateisystems** – Mit den Befehlen `zfs snapshot` und `zfs rollback` können Sie auf einfache Weise Kopien von Dateisystemen erstellen und aus diesen bei Bedarf Dateisysteme auf einen früheren Zustand zurücksetzen. Sie können die Lösung beispielsweise verwenden, wenn Sie ein Dateisystem bzw. Dateien aus einer früheren Dateisystemversion wiederherstellen möchten.

  Weitere Informationen zum Erstellen von Snapshots und Wiederherstellen einer früheren Version aus einem Snapshot finden Sie unter „Überblick über ZFS-Snapshots“ auf Seite 225.
- Sichern von Snapshots – Mit den Befehlen `zfs send` und `zfs receive` können Sie ZFS-Snapshots senden und empfangen. Sie können inkrementelle Änderungen zwischen Snapshots sichern, Dateien können jedoch nicht einzeln wiederhergestellt werden. Stattdessen muss der gesamte Dateisystem-Snapshot wiederhergestellt werden. Diese Befehle stellen keine vollständige Sicherungslösung für Ihre ZFS-Daten dar.
- **Replikation über Netzwerk** – Mit den Befehlen `zfs send` und `zfs receive` können Sie Dateisysteme von einem System auf ein anderes kopieren. Dieser Vorgang unterscheidet sich von herkömmlichen Praktiken bei Software zur Datenträgerverwaltung, die Datenspeichergeräte über WAN spiegeln. Dafür ist keine spezielle Konfiguration bzw. Hardware erforderlich. Der Vorteil der Replikation eines ZFS-Dateisystems besteht darin, dass das betreffende Dateisystem in einem Speicher-Pool eines anderen Systems neu erstellt werden kann und Sie für diesen neuen Pool verschiedene Replikationsmethoden wie z. B. RAID-Z angeben können, die Dateisystemdaten aber gleich bleiben.
- Archivierungsdienstprogramme – Speichern von ZFS-Daten mit Archivierungsdienstprogrammen wie `tar`, `cpio` und `pax` oder Datensicherungssoftware von Drittherstellern. Derzeit konvertieren `tar` und `cpio` die NFSv4-basierten Zugriffssteuerungslisten korrekt, `pax` hingegen nicht.

Sichern von ZFS-Daten mit anderen Softwarepaketen zur Erstellung von Sicherungskopien

Neben den Befehlen `zfs send` und `zfs receive` können Sie zum Sichern von ZFS-Dateien auch Archivierungsdienstprogramme wie z. B. `tar` und `cpio` verwenden. Diese Dienstprogramme sichern ZFS-Dateiattribute und -Zugriffssteuerungslisten und können diese auch wiederherstellen. Überprüfen Sie die entsprechenden Optionen der Befehle `tar` und `cpio`.

Aktuelle Informationen zu Problemen im Zusammenhang mit ZFS und Sicherungssoftware von Drittherstellern finden Sie in den Solaris 10-Versionshinweisen oder den ZFS FAQs unter:

`http://hub.opensolaris.org/bin/view/Community+Group+zfs/faq/#backupsoftware`

Senden von ZFS-Snapshots

Der Befehl `zfs send` dient zum Senden der Kopie eines Snapshot-Datenstroms und zum Empfangen des Snapshot-Datenstroms in einem anderen Pool auf demselben System oder in einem anderen Pool auf einem anderen System, das zur Aufbewahrung von Sicherungsdaten verwendet wird. Zum Senden des Snapshot-Datenstroms an einen anderen Pool auf demselben System verwenden Sie beispielsweise folgende Syntax:

```
# zfs send tank/data@snap1 | zfs recv spool/ds01
```

Sie können `zfs recv` als Aliasnamen für den Befehl `zfs receive` verwenden.

Wenn Sie den Snapshot-Datenstrom an ein anderes System senden, setzen Sie für die Ausgabe von `zfs send` mit dem Befehl `ssh` eine Pipeline. Beispiel:

```
host1# zfs send tank/dana@snap1 | ssh host2 zfs recv newtank/dana
```

Wenn Sie einen vollständige Datenstrom senden, darf das Zieldateisystem nicht vorhanden sein.

Sie können inkrementelle Daten mit der Option `i` des Befehls `-zfs send` senden. Beispiel:

```
host1# zfs send -i tank/dana@snap1 tank/dana@snap2 | ssh host2 zfs recv newtank/dana
```

Bitte beachten Sie, dass das erste Argument (`snap1`) der frühere und das zweite Argument (`snap2`) der spätere Snapshot ist. In diesem Fall muss das Dateisystem `newtank/dana` bereits vorhanden sein, damit die inkrementellen Daten empfangen werden können.

Die inkrementelle Quelle *snap1* kann als letzte Komponente des Snapshot-Namens angegeben werden. Dies bedeutet, dass Sie nach dem Zeichen `@` für *snap1* nur den Namen angeben müssen, von dem angenommen wird, dass er zum gleichen System wie *snap2* gehört. Beispiel:

```
host1# zfs send -i snap1 tank/dana@snap2 > ssh host2 zfs recv newtank/dana
```

Diese Kurzsyntax entspricht der im vorherigen Beispiel demonstrierten inkrementellen Syntax.

Die folgende Meldung wird angezeigt, wenn Sie versuchen, aus einem anderen Dateisystem (`snapshot1`) einen inkrementellen Datenstrom zu erzeugen.

```
cannot send 'pool/fs@name': not an earlier snapshot from the same fs
```

Wenn mehrere Kopien gespeichert werden sollen, kann eine Komprimierung der Datenstrominstanz des ZFS-Snapshots mithilfe des Befehls `gzip` nützlich sein. Beispiel:

```
# zfs send pool/fs@snap | gzip > backupfile.gz
```

Empfangen von ZFS-Snapshots

Beim Empfangen von Datensystem-Snapshots sollten Sie folgenden wichtigen Punkte beachten:

- Sowohl der Snapshot als auch das Dateisystem werden empfangen.
- Das Dateisystem und alle untergeordneten Dateisysteme werden ausgehängt.
- Während des Empfangs kann auf die betreffenden Dateisysteme nicht zugegriffen werden.
- Das ursprüngliche Dateisystem, das empfangen werden soll, darf bei der Übertragung nicht vorhanden sein.
- Wenn der Dateisystemname bereits vorhanden ist, können Sie das Dateisystem mit dem Befehl `zfs rename` umbenennen.

Beispiel:

```
# zfs send tank/gozer@0830 > /bkups/gozer.083006
# zfs receive tank/gozer2@today < /bkups/gozer.083006
# zfs rename tank/gozer tank/gozer.old
# zfs rename tank/gozer2 tank/gozer
```

Wenn Sie am Zieldateisystem eine Änderung vornehmen und danach einen weiteren inkrementellen Snapshot senden möchten, müssen Sie zunächst den vorherigen Zustand des Zieldateisystems wiederherstellen.

Betrachten wir das folgende Beispiel. Zunächst ändern Sie das Dateisystem wie folgt:

```
host2# rm newtank/dana/file.1
```

Dann senden Sie einen weiteren inkrementellen Snapshot (`tank/dana@snap3`). Sie müssen jedoch erst den vorherigen Zustand des Zieldateisystems wiederherstellen, damit es den neuen inkrementellen Snapshot empfangen kann. Sie können den Wiederherstellungsschritt aber auch mithilfe der Option `-F` überspringen. Beispiel:

```
host1# zfs send -i tank/dana@snap2 tank/dana@snap3 | ssh host2 zfs recv -F newtank/dana
```

Beim Empfang eines inkrementellen Snapshots muss das Zieldateisystem bereits vorhanden sein.

Wenn Sie am Dateisystem Änderungen vornehmen und den vorherigen Zustand des Zieldateisystems nicht wiederherstellen, sodass es den neuen inkrementellen Snapshot empfangen kann, oder Sie die Option `-F` nicht verwenden, wird eine Meldung wie die folgende angezeigt:

```
host1# zfs send -i tank/dana@snap4 tank/dana@snap5 | ssh host2 zfs recv newtank/dana
cannot receive: destination has been modified since most recent snapshot
```

Bevor das Ausführen der Option `-F` als erfolgreich gemeldet wird, werden die folgenden Überprüfungen durchgeführt:

- Wenn der letzte Snapshot nicht mit der inkrementellen Quelle identisch ist, wird weder die Wiederherstellung des früheren Zustands noch der Empfang abgeschlossen, und es wird eine Fehlermeldung angezeigt.
- Wenn Sie versehentlich den Namen eines anderen Dateisystems angeben, der mit dem inkrementellen Quellparameter des Befehls `zfs receive` nicht übereinstimmt, wird weder die Wiederherstellung des früheren Zustands noch der Empfang abgeschlossen, und es wird die folgende Fehlermeldung angezeigt:

```
cannot send 'pool/fs@name': not an earlier snapshot from the same fs
```

Senden und Empfangen komplexer ZFS-Snapshot-Datenströme

In diesem Abschnitt wird das Senden und Empfangen komplexerer Snapshot-Datenströme mit den Befehlen `zfs send -I` und `-R` beschrieben.

Beachten Sie beim Senden und Empfangen von komplexen ZFS-Snapshot-Datenströmen Folgendes:

- Verwenden Sie `zfs send` mit der Option `-I`, um alle inkrementellen Datenströme aus einem Snapshot an einen kumulativen Snapshot zu senden. Sie können mithilfe dieser Option aber auch einen inkrementellen Datenstrom aus dem ursprünglichen Snapshot senden, um einen Klon zu erstellen. Der ursprüngliche Snapshot muss auf der Empfangsseite bereits vorhanden sein, damit der inkrementelle Datenstrom angenommen werden kann.
- Mit `zfs send` und der Option `-R` senden Sie einen Replikationsdatenstrom aller untergeordneten Dateisysteme. Nach dem Empfang des Replikationsdatenstroms werden alle Eigenschaften, Snapshots, abhängigen Dateisysteme und Klone beibehalten.
- Verwenden Sie beide Optionen, um einen inkrementellen Replikationsdatenstrom zu senden.
  - Änderungen an Eigenschaften werden beibehalten, ebenso wie Namensänderungen von Snapshots und Dateisystemen und Löschvorgänge.
  - Wenn `zfs recv -F` beim Empfang des Replikationsdatenstroms nicht angegeben ist, werden Löschvorgänge von Datasets ignoriert. In diesem Fall behält die Syntax `zfs recv -F` auch die Bedeutung *Bei Bedarf Rollback* bei.
  - Wie in anderen Fällen (außer `zfs send -R`) mit `- i` oder `-I` werden bei Verwendung von `-I` alle Snapshots zwischen snapA und snapD gesendet. Bei Verwendung von `-i` wird nur `snapD` (für sämtliche untergeordneten Objekte) gesendet.
- Der Empfang dieser mithilfe des Befehls `zfs send` gesendeten neuen Datenströme setzt voraus, dass auf dem empfangenden System eine Softwareversion ausgeführt wird, die in der Lage ist, diese Datenströme zu senden. Die Version des Datenstroms wird inkrementiert.

Es ist jedoch möglich, auf Datenströme aus älteren Pool-Versionen über neuere Softwareversionen zuzugreifen. So können Sie beispielsweise Datenströme, die mit den neueren Optionen erstellt wurden, an und aus Pools der Version 3 senden. Zum Empfangen eines mit den neueren Optionen gesendeten Datenstroms muss jedoch aktuelle Software ausgeführt werden.

BEISPIEL 7–1 Senden und Empfangen komplexer ZFS-Snapshot-Datenströme

Eine Gruppe inkrementeller Snapshots lässt sich mithilfe von `zfs send` und der Option `-I` zu einem Snapshot kombinieren. Beispiel:

```
# zfs send -I pool/fs@snapA pool/fs@snapD > /snaps/fs@all-I
```

Anschließend entfernen Sie `snapB`, `snapC` und `snapD`.

```
# zfs destroy pool/fs@snapB
# zfs destroy pool/fs@snapC
# zfs destroy pool/fs@snapD
```

Um den kombinierten Snapshot zu empfangen, verwenden Sie den folgenden Befehl.

```
# zfs receive -d -F pool/fs < /snaps/fs@all-I
# zfs list
NAME                      USED  AVAIL  REFER  MOUNTPOINT
pool                      428K  16.5G    20K  /pool
pool/fs                    71K  16.5G    21K  /pool/fs
pool/fs@snapA              16K      -  18.5K  -
pool/fs@snapB              17K      -    20K  -
pool/fs@snapC              17K      -  20.5K  -
pool/fs@snapD                0      -    21K  -
```

Außerdem können Sie mit `zfs send -I` einen Snapshot und einen Klon-Snapshot zu einem kombinierten Dataset verbinden. Beispiel:

```
# zfs create pool/fs
# zfs snapshot pool/fs@snap1
# zfs clone pool/fs@snap1 pool/clone
# zfs snapshot pool/clone@snapA
# zfs send -I pool/fs@snap1 pool/clone@snapA > /snaps/fsclonesnap-I
# zfs destroy pool/clone@snapA
# zfs destroy pool/clone
# zfs receive -F pool/clone < /snaps/fsclonesnap-I
```

Mit dem Befehl `zfs send -R` können Sie ein ZFS-Dateisystem und alle untergeordneten Dateisysteme bis hin zum benannten Snapshot replizieren. Nach dem Empfang dieses Datenstroms werden alle Eigenschaften, Snapshots, abhängigen Dateisysteme und Klone beibehalten.

Im folgenden Beispiel werden Snapshots für Benutzerdateisysteme erstellt. Es wird ein Replikationsdatenstrom für alle Benutzer-Snapshots erstellt. Anschließend werden die ursprünglichen Dateisysteme und Snapshots gelöscht und wiederhergestellt.

BEISPIEL 7–1 Senden und Empfangen komplexer ZFS-Snapshot-Datenströme *(Fortsetzung)*

```
# zfs snapshot -r users@today
# zfs list
NAME                USED  AVAIL  REFER  MOUNTPOINT
users               187K  33.2G    22K  /users
users@today            0      -    22K  -
users/user1          18K  33.2G    18K  /users/user1
users/user1@today      0      -    18K  -
users/user2          18K  33.2G    18K  /users/user2
users/user2@today      0      -    18K  -
users/user3          18K  33.2G    18K  /users/user3
users/user3@today      0      -    18K  -
# zfs send -R users@today > /snaps/users-R
# zfs destroy -r users
# zfs receive -F -d users < /snaps/users-R
# zfs list
NAME                USED  AVAIL  REFER  MOUNTPOINT
users               196K  33.2G    22K  /users
users@today            0      -    22K  -
users/user1          18K  33.2G    18K  /users/user1
users/user1@today      0      -    18K  -
users/user2          18K  33.2G    18K  /users/user2
users/user2@today      0      -    18K  -
users/user3          18K  33.2G    18K  /users/user3
users/user3@today      0      -    18K  -
```

Im folgenden Beispiel wird der Befehl `zfs send -R` verwendet, um das Dataset `users` und seine untergeordneten Objekte zu replizieren und den replizierten Datenstrom an einen anderen Pool, `users2`, zu senden.

```
# zfs create users2 mirror c0t1d0 c1t1d0
# zfs receive -F -d users2 < /snaps/users-R
# zfs list
NAME                 USED  AVAIL  REFER  MOUNTPOINT
users                224K  33.2G    22K  /users
users@today             0      -    22K  -
users/user1           33K  33.2G    18K  /users/user1
users/user1@today     15K      -    18K  -
users/user2           18K  33.2G    18K  /users/user2
users/user2@today       0      -    18K  -
users/user3           18K  33.2G    18K  /users/user3
users/user3@today       0      -    18K  -
users2               188K  16.5G    22K  /users2
users2@today            0      -    22K  -
users2/user1          18K  16.5G    18K  /users2/user1
users2/user1@today      0      -    18K  -
users2/user2          18K  16.5G    18K  /users2/user2
users2/user2@today      0      -    18K  -
users2/user3          18K  16.5G    18K  /users2/user3
users2/user3@today      0      -    18K  -
```

Replikation von ZFS-Daten über das Netzwerk

Mit den Befehlen `zfs send` und `zfs recv` können Sie eine Datenstrominstanz über das Netzwerk von einem System auf ein anderes kopieren. Beispiel:

```
# zfs send tank/cindy@today | ssh newsys zfs recv sandbox/restfs@today
```

Mithilfe des Befehls wird der Snapshot `tank/cindy@today` gesendet und vom Dateisystem `sandbox/restfs` empfangen. Außerdem wird ein `restfs@today`-Snapshot für das System `newsys` erstellt. In diesem Beispiel wird auf dem entfernten System `ssh` verwendet.



KAPITEL 8

Schützen von Oracle Solaris ZFS-Dateien mit Zugriffssteuerungslisten

Dieses Kapitel enthält Informationen zum Arbeiten mit Zugriffssteuerungslisten. Diese Listen dienen zum Schutz von ZFS-Dateien. Zugriffssteuerungslisten definieren im Vergleich zu UNIX-Standardzugriffsrechten feiner abgestimmte Zugriffsrechte.

Dieses Kapitel enthält die folgenden Abschnitte:

Neues Solaris-Modell für Zugriffssteuerungslisten

Frühere Versionen des Betriebssystems Solaris unterstützten eine auf dem POSIX-Entwurf basierende Implementierung von Zugriffssteuerungslisten. POSIX-basierte Zugriffssteuerungslisten dienen zum Schutz von UFS-Dateien und werden von NFS-Versionen vor NFSv4 verwendet.

Seit der Einführung von NFSv4 unterstützt ein neues Modell für Zugriffssteuerungslisten vollständig die Interoperabilität, die NFSv4 für die Kommunikation zwischen UNIX-Clients und anderen Clients bietet. Die neue, in der NFSv4-Spezifikation definierte Implementierung von Zugriffssteuerungslisten bietet eine reichhaltigere Semantik, die auf NT-basierten Zugriffssteuerungslisten beruht.

Es folgen die wichtigsten Unterschiede des neuen Zugriffssteuerungslistenmodells:

- Das neue Zugriffssteuerungslistenmodell basiert auf der NFSv4-Spezifikation und ist den NT-Zugriffssteuerungslistenmodellen ähnlich.
- Das neue Modell bietet feiner abgestimmte Zugriffsrechte. Weitere Informationen finden Sie in Tabelle 8–2.

- Zugriffssteuerungslisten werden mit den Befehlen `chmod` und `ls` anstatt `setfacl` und `getfacl` eingestellt und angezeigt.
- Das neue Modell bietet eine reichhaltigere Vererbungssemantik zum Festlegen der Weitergabe von Zugriffsrechten von über- an untergeordnete Verzeichnisse usw. Weitere Informationen dazu finden Sie in „Vererbung von Zugriffssteuerungslisten“ auf Seite 248.

Beide Zugriffssteuerungslistenmodelle bieten eine feiner abstimmbare Kontrolle von Zugriffsrechten als die Standardzugriffsrechte. Ähnlich wie POSIX-basierten Zugriffssteuerungslisten bestehen auch die neuen Zugriffssteuerungslisten aus mehreren Zugriffssteuerungseinträgen.

POSIX-basierte Zugriffssteuerungslisten definieren mithilfe eines einzigen Eintrags, welche Zugriffsrechte zulässig und unzulässig sind. Das neue Zugriffssteuerungslistenmodell besitzt zwei Arten von Zugriffssteuerungseinträgen, die sich auf die Überprüfung von Zugriffsrechten auswirken: `ALLOW` (Erlauben) und `DENY` (Verweigern). Demzufolge können Sie nicht aus einem einzigen Zugriffssteuerungseintrag schließen, ob die in diesem Zugriffssteuerungseintrag nicht definierten Zugriffsrechte zulässig sind oder nicht.

Die Konvertierung zwischen NFSv4-basierten und POSIX-basierten Zugriffssteuerungslisten geschieht wie folgt:

- Bei der Verwendung von Dienstprogrammen, die Zugriffsrechte überprüfen (z. B. die Befehle `cp`, `mv`, `tar`, `cpio` oder `rcp`), werden die POSIX-basierten Zugriffssteuerungslisten in die entsprechenden NFSv4-basierten Zugriffssteuerungslisten konvertiert, damit UFS-Dateien mit Zugriffssteuerungslisten in ein ZFS-Dateisystem transferiert werden können.
- Einige NFSv4-basierte Zugriffssteuerungslisten werden in POSIX-basierte Zugriffssteuerungslisten konvertiert. Wenn eine NFSv4-basierte Zugriffssteuerungsliste nicht in eine POSIX-basierte Zugriffssteuerungsliste konvertiert werden kann, wird eine Meldung angezeigt, die der folgenden ähnlich ist:

  ```
  # cp -p filea /var/tmp
  cp: failed to set acl entries on /var/tmp/filea
  ```
- Wenn Sie auf einem System, auf dem die neueste Solaris-Version installiert ist, mit `tar` oder `cpio` ein UFS-Archiv mit der Option zur Beibehaltung der Zugriffssteuerungsliste (`tar -p` bzw. `cpio -P`) erstellen, gehen beim Extrahieren des Archivs auf einem System, auf dem eine frühere Solaris-Version installiert ist, die Zugriffssteuerungslisten verloren.

  Alle Dateien werden mit den ordnungsgemäßen Dateimodi extrahiert, aber die Zugriffssteuerungslisten werden ignoriert.
- Der Befehl `ufsrestore` ermöglicht es, Daten in einem ZFS-Dateisystem wiederherzustellen. Wenn die Originaldaten POSIX-basierte Zugriffssteuerungslisten enthalten, werden diese in NFSv4-basierte Zugriffssteuerungslisten konvertiert.
- Wenn Sie versuchen, eine NFSv4-basierte Zugriffssteuerungsliste für eine UFS-Datei festzulegen, wird eine Meldung wie die folgende angezeigt:

  ```
  chmod: ERROR: ACL type's are different
  ```

- Wenn Sie versuchen, eine POSIX-basierte Zugriffssteuerungsliste für eine ZFS-Datei festzulegen, wird eine Meldung wie die folgende angezeigt:

```
# getfacl filea
File system doesn't support aclent_t style ACL's.
See acl(5) for more information on Solaris ACL support.
```

Informationen zu Einschränkungen mit Zugriffssteuerungslisten und Backup-Software finden Sie unter „Sichern von ZFS-Daten mit anderen Softwarepaketen zur Erstellung von Sicherungskopien“ auf Seite 235.

Syntaxbeschreibungen zum Setzen von Zugriffssteuerungslisten

Es gibt zwei grundlegende Zugriffssteuerungslistenformate:

Syntax zum Setzen gewöhnlicher Zugriffssteuerungslisten

Eine Zugriffssteuerungsliste ist insofern *gewöhnlich*, als sie nur die herkömmlichen UNIX-Einträge `owner/group/other` repräsentiert.

`chmod [Optionen] A[Index]{+|=}owner@ |group@ |everyone@:` *Zugriffsrechte*/...[:*Vererbungsflags*]: `deny | allow` *Datei*

`chmod [Optionen] A-owner@, group@, everyone@:` *Zugriffsrechte*/...[:*Vererbungsflags*]: `deny | allow` *Datei* ...

`chmod [Optionen] A[Index]-` *Datei*

Syntax zum Setzen komplexerer Zugriffssteuerungslisten

`chmod [Optionen] A[Index]{+|=}user|group:name:`*Zugriffsrechte* /...[:*Vererbungsflags*]`:deny | allow` *Datei*

`chmod [Optionen] A-user|group:name:`*Zugriffsrechte* /...[:*Vererbungsflags*]`:deny | allow` *Datei* ...

`chmod [Optionen] A[Index]-` *Datei*

`owner@`, `group@`, `everyone@`
: Der *Zugriffssteuerungslisten-Eintragstyp* für die gewöhnliche Zugriffssteuerungslistensyntax. Eine Beschreibung der Zugriffssteuerungslisten-Eintragstypen finden Sie in Tabelle 8–1.

user oder group:Zugriffssteuerungslisten-Eintragstyp-ID = Benutzername oder Gruppenname
: Der *Zugriffssteuerungslisten-Eintragstyp* für die explizite Zugriffssteuerungslistensyntax. `Der` *Zugriffssteuerungslisten-Eintragstyp* „user” bzw. ”group” muss auch die *Zugriffssteuerungslisteneintrags-ID*, den *Benutzernamen* oder den *Gruppennamen* enthalten. Eine Beschreibung der Zugriffssteuerungslisten-Eintragstypen finden Sie in Tabelle 8–1.

Zugriffsrechte/…/
Die Zugriffsrechte, die gewährt bzw. verweigert werden. Eine Beschreibung von Zugriffsrechten für Zugriffssteuerungslisten finden Sie in Tabelle 8–2.

Vererbungsflags
Eine optionale Liste von Vererbungsflags von Zugriffssteuerungslisten. Eine Beschreibung der Vererbungsflags von Zugriffssteuerungslisten finden Sie in Tabelle 8–3.

`deny | allow`
Legt fest, ob Zugriffsrechte gewährt oder verweigert werden.

Im folgenden Beispiel spielt die *Zugriffssteuerungslisten-Eintragstyp-ID* keine Rolle:

```
group@:write_data/append_data/execute:deny
```

Das folgende Beispiel enthält eine *Zugriffssteuerungslisteneintrags-ID*, da ein spezifischer Benutzer (*Zugriffssteuerungslisten-Eintragstyp*) in der Zugriffssteuerungsliste enthalten ist.

```
0:user:gozer:list_directory/read_data/execute:allow
```

Ein Zugriffssteuerungslisteneintrag sieht ungefähr wie folgt aus:

```
2:group@:write_data/append_data/execute:deny
```

Die **2** bzw. die *Index-ID* in diesem Beispiel identifiziert den Zugriffssteuerungslisteneintrag in der größeren Zugriffssteuerungsliste, in der für Eigentümer, spezifische UIDs, Gruppen und allgemeine Zugriffsrechte mehrere Einträge vorhanden sein können. Sie können die *Index-ID* mit dem Befehl `chmod` angeben und somit festlegen, welchen Teil der Zugriffssteuerungsliste Sie ändern möchten. Sie können beispielsweise Index-ID 3 als `A3` im Befehl `chmod` angeben (siehe folgendes Beispiel):

```
chmod A3=user:venkman:read_acl:allow filename
```

Zugriffssteuerungslisten-Eintragstypen, die Eigentümer, Gruppen und andere Parameter in Zugriffssteuerungslisten darstellen, sind in der folgenden Tabelle beschrieben.

TABELLE 8–1 Zugriffssteuerungslisten- Eintragstypen

Zugriffssteuerungslisten-Eintragstyp	Beschreibung
`owner@`	Das Zugriffsrecht, das dem Eigentümer des Objekts gewährt wird.
`group@`	Das Zugriffsrecht, das der Eigentümergruppe des Objekts gewährt wird.
`everyone@`	Der Zugriff, der allen anderen Benutzern oder Gruppen gewährt wird, denen keine anderen Zugriffssteuerungslisteneinträge entsprechen.

TABELLE 8–1 Zugriffssteuerungslisten- Eintragstypen *(Fortsetzung)*

Zugriffssteuerungslisten-Eintragstyp	Beschreibung
`user`	Das Zugriffsrecht, das einem zusätzlichen (und durch den Benutzernamen anzugebenden) Benutzer des Objekts gewährt wird. Dieser Eintrag muss die *Zugriffssteuerungslisteneintrags-ID* enthalten, die wiederum den betreffenden *Benutzernamen* oder die *Benutzer-ID* enthält. Wenn es sich bei diesem Wert um keine gültige nummerische Benutzer-ID oder keinen *Benutzername* handelt, ist der Zugriffssteuerungslisten-Eintragstyp ungültig.
`group`	Das Zugriffsrecht, das einer zusätzlichen (und durch den Gruppennamen anzugebenden) Benutzergruppe des Objekts gewährt wird. Dieser Eintrag muss die *Zugriffssteuerungslisteneintrags-ID* enthalten, die wiederum den betreffenden *Gruppennamen* oder die *Gruppen-ID* enthält. Wenn es sich bei diesem Wert um keine gültige nummerische Gruppen-ID oder keinen *Gruppenname* handelt, ist der Zugriffssteuerungslisten-Eintragstyp ungültig.

In der folgenden Tabelle sind die Zugriffsrechte für Zugriffssteuerungslisten beschrieben.

TABELLE 8–2 Zugriffssteuerungslisten-Zugriffsrechte

Zugriffsrecht	Kurzform	Beschreibung
`add_file`	`w`	Berechtigung zum Hinzufügen neuer Dateien zu einem Verzeichnis.
`add_subdirectory`	`p`	Berechtigung zum Erstellen von Unterverzeichnissen in einem Verzeichnis.
`append_data`	`p`	Platzhalter. Gegenwärtig nicht implementiert.
`delete`	`d`	Berechtigung zum Löschen einer Datei.
`delete_child`	`D`	Berechtigung zum Löschen einer Datei oder eines Verzeichnisses in einem Verzeichnis.
`execute`	`x`	Berechtigung zum Ausführen einer Datei bzw. Durchsuchen eines Verzeichnisses.
`list_directory`	`r`	Berechtigung zum Auflisten des Inhalts eines Verzeichnisses.
`read_acl`	`c`	Berechtigung zum Lesen der Zugriffssteuerungsliste (`ls`).
`read_attributes`	`a`	Berechtigung zum Lesen grundlegender (nicht zu Zugriffssteuerungslisten gehörender) Attribute einer Datei. Grundlegende Attribute sind Attribute auf der stat-Ebene. Durch Setzen dieses Zugriffsmaskenbits kann eine Entität `ls`(1) und `stat`(2) ausführen.
`read_data`	`r`	Berechtigung zum Lesen des Inhalts einer a Datei.

TABELLE 8–2 Zugriffssteuerungslisten-Zugriffsrechte *(Fortsetzung)*

Zugriffsrecht	Kurzform	Beschreibung
read_xattr	R	Berechtigung zum Lesen der erweiterten Attribute einer Datei bzw. Durchsuchen des Verzeichnisses erweiterter Dateiattribute.
synchronize	s	Platzhalter. Gegenwärtig nicht implementiert.
write_xattr	W	Berechtigung zum Erstellen erweiterter Attribute bzw. Schreiben in das Verzeichnis erweiterter Attribute. Durch Gewähren dieses Zugriffsrechtes kann ein Benutzer für eine Datei ein Verzeichnis erweiterter Attribute erstellen. Die Zugriffsrechte der Attributdatei legen das Zugriffsrecht des Benutzers auf das Attribut fest.
write_data	w	Berechtigung zum Ändern oder Ersetzens des Inhalts einer Datei.
write_attributes	W	Berechtigung zum Setzen der Zeitmarken einer Datei bzw. eines Verzeichnisses auf einen beliebigen Wert.
write_acl	C	Berechtigung zum Schreiben bzw. Ändern der Zugriffsteuerungsliste mithilfe des Befehls chmod.
write_owner	o	Berechtigung zum Ändern des Eigentümers bzw. der Gruppe einer Datei bzw. Berechtigung zum Ausführen der Befehle chown oder chgrp für die betreffende Datei. Berechtigung zum Übernehmen der Eigentümerschaft einer Datei bzw. zum Ändern der ihrer Gruppe, zu der der betreffende Benutzer gehört. Wenn Sie die Benutzer- bzw. Gruppeneigentümerschaft auf einen beliebigen Benutzer bzw. eine beliebige Gruppe setzen wollen, ist das Zugriffsrecht PRIV_FILE_CHOWN erforderlich.

Vererbung von Zugriffssteuerungslisten

Der Zweck der Vererbung von Zugriffssteuerungslisten besteht darin, dass neu erstellte Dateien bzw. Verzeichnisse die vorgesehenen Zugriffssteuerungslisten erben, ohne dass die vorhandenen Zugriffsrechte des übergeordneten Verzeichnisses ignoriert werden.

Standardmäßig werden Zugriffssteuerungslisten nicht weitergegeben. Wenn Sie für ein Verzeichnis eine komplexe Zugriffssteuerungsliste setzen, wird diese nicht an untergeordnete Verzeichnisse vererbt. Sie müssen die Vererbung einer Zugriffssteuerungsliste für Dateien bzw. Verzeichnisse explizit angeben.

In der folgenden Tabelle sind die optionalen Vererbungsflags aufgeführt.

TABELLE 8–3 Vererbungsflags von Zugriffssteuerungslisten

Vererbungsflag	Kurzform	Beschreibung
`file_inherit`	`f`	Die Zugriffssteuerungsliste wird vom übergeordneten Verzeichnis nur an die Dateien des betreffenden Verzeichnisses vererbt.
`dir_inherit`	`d`	Die Zugriffssteuerungsliste wird vom übergeordneten Verzeichnis nur an die Unterverzeichnisse des betreffenden Verzeichnisses vererbt.
`inherit_only`	`i`	Die Zugriffssteuerungsliste wird vom übergeordneten Verzeichnis vererbt, gilt jedoch nur für neu erstellte Dateien bzw. Unterverzeichnisse, aber nicht für das betreffende Verzeichnis selbst. Für dieses Flag ist das Flag `file_inherit` bzw. `dir_inherit` (oder beide) erforderlich. Diese legen fest, was vererbt wird.
`no_propagate`	`n`	Die Zugriffssteuerungsliste wird vom übergeordneten Verzeichnis an die erste Hierarchieebene des betreffenden Verzeichnisses und nicht an untergeordnete Ebenen vererbt. Für dieses Flag ist das Flag `file_inherit` bzw. `dir_inherit` (oder beide) erforderlich. Diese legen fest, was vererbt wird.
-	entf.	Zugriff nicht gewährt.

Darüber hinaus können Sie mithilfe der Dateisystemeigenschaft `aclinherit` für das Dateisystem mehr oder weniger strenge Vererbungsrichtlinien für Zugriffssteuerungslisten festlegen. Weitere Informationen finden Sie im folgenden Abschnitt.

Eigenschaften von Zugriffssteuerungslisten

Das ZFS-Dateisystem enthält zwei Eigenschaften für Zugriffssteuerungslisten.

- `aclinherit` – Diese Eigenschaft legt das Verhalten der Vererbung von Zugriffssteuerungslisten fest und kann die folgenden Werte annehmen:
  - `discard` – Für neu erstellte Objekte werden beim Erstellen von Dateien und Verzeichnissen keine Zugriffssteuerungslisteneinträge vererbt. Die Zugriffssteuerungsliste der neuen Datei bzw. des neuen Verzeichnisses entspricht dem Zugriffsrechten dieser Datei bzw. dieses Verzeichnisses.
  - `noallow` – Bei neuen Objekten werden nur vererbbare Zugriffssteuerungslisteneinträge mit Zugriffstyp `deny` vererbt.
  - `restricted` – Für neu erstellte Objekte werden beim Vererben von Zugriffssteuerungslisteneinträgen die Zugriffsrechte `write_owner` und `write_acl` entfernt.

- passthrough – Ist der Eigenschaftswert auf passthrough gesetzt, werden die Dateien mithilfe von Zugriffsrechten erstellt, die durch die vererbbaren Zugriffssteuerungseinträge festgelegt wurden. Wenn keine Zugriffsrechte für die vererbbaren Zugriffssteuerungseinträge vorhanden sind, werden Zugriffsrechte gesetzt, die mit der Forderung der Anwendung übereinstimmen.
- passthrough-x – Dieser Eigenschaftswert hat die gleiche Semantik wie passthrough, mit der Ausnahme, dass bei Aktivierung von passthrough-x Dateien mit der Ausführungsberechtigung (x) erstellt werden, jedoch nur, wenn die Ausführungsberechtigung im Dateierstellungsmodus und in einer vererbbaren Zugriffssteuerungsliste festgelegt wurde, die sich auf den Modus auswirkt.

Der Standardwert für die Eigenschaft aclinherit ist beschränkt.

- aclmode – Diese Eigenschaft ändert das Zugriffssteuerungslistenverhalten, wenn eine Datei neu erstellt wird oder die Datei- bzw. Verzeichniszugriffsrechte vom Befehl chmod geändert werden, und kann die folgenden Werte annehmen:
  - discard – Alle Zugriffssteuerungslisteneinträge außer denen, die den Modus der Datei bzw. des Verzeichnisses definieren, werden entfernt.
  - groupmask – Zugriffssteuerungslisten-Zugriffsrechte für Benutzer bzw. Gruppen werden eingeschränkt, sodass sie nicht größer werden als durch die Gruppenzugriffsrechte festgelegt, es sei denn, es handelt sich um einen Benutzereintrag mit der gleichen Benutzer-ID wie der Eigentümer der Datei bzw. des Verzeichnisses. Dann werden die Zugriffssteuerungslisten-Zugriffsrechte eingeschränkt, sodass sie nicht größer werden als durch die Zugriffsrechte des Eigentümers gesetzt.
  - passthrough – Während eines chmod-Vorgangs werden Zugriffssteuerungseinträge außer owner@, group@ oder everyone@ nicht geändert. Zugriffssteuerungseinträge mit owner@, group@ oder everyone@ werden deaktiviert, um den Dateimodus wie durch den chmod-Vorgang gefordert zu setzen.

  Der Standardwert für aclmode ist groupmask.

Setzen von Zugriffssteuerungslisten an ZFS-Dateien

In ZFS bestehen Zugriffssteuerungslisten aus Zugriffssteuerungslisteneinträgen. ZFS bietet ein *reines* Zugriffssteuerungslistenmodell, in dem alle Dateien eine Zugriffssteuerungsliste besitzen. Normalerweise sind solche Zugriffssteuerungslisten *gewöhnlich*, da sie nur die herkömmlichen UNIX-Einträge owner/group/other repräsentieren.

Wenn Sie also die Zugriffsrechte einer Datei ändern, wird die entsprechende Zugriffssteuerungsliste automatisch aktualisiert. Wenn Sie eine komplexe Zugriffssteuerungsliste entfernen, die einem Benutzer Zugriff auf eine Datei bzw. ein Verzeichnis gewährt, kann dieser Benutzer unter Umständen noch immer Zugriff auf diese Datei bzw. dieses Verzeichnis haben, da die Zugriffsrecht-Bits dieser Datei bzw. dieses

Verzeichnisses einer Gruppe oder allen Benutzern Zugriff gewähren. Alle Entscheidungen in Sachen Zugriffskontrolle werden von den Zugriffsrechten der Zugriffssteuerungsliste dieser Datei bzw. dieses Verzeichnisses festgelegt.

Es folgen die grundlegenden Regeln für den Zugriff auf ZFS-Dateien mithilfe von Zugriffssteuerungslisten:

- ZFS verarbeitet Zugriffssteuerungslisteneinträge in der Reihenfolge, wie sie in der Zugriffssteuerungsliste aufgeführt sind (d. h. von oben nach unten).
- Es werden nur Zugriffssteuerungslisteneinträge berücksichtigt, deren Benutzer-ID mit der ID des Zugriff anfordernden Benutzer übereinstimmt.
- Wenn ein Zugriffsrecht einmal gewährt wurde, kann es durch nachfolgende Zugriffssteuerungslisteneinträge in der gleichen Zugriffssteuerungsliste nicht mehr rückgängig gemacht werden.
- Eigentümern von Dateien wird das Zugriffsrecht `write_acl` auch dann bedingungslos gewährt, wenn es in der Zugriffssteuerungsliste explizit verweigert wird. Nicht angegebene Zugriffsrechte werden verweigert.

  Falls Zugriffsrechte verweigert werden bzw. Angaben zum Zugriff auf eine Datei fehlen, legen die Zugriffsrechte des betreffenden untergeordneten Systems fest, welche Rechte dem Eigentümer einer Datei bzw. dem Superuser gewährt werden. Dadurch wird verhindert, dass der Zugriff für Dateieigentümernicht gesperrt wird und Superuser Dateien aus Gründen der Wiederherstellung ändern können.

Wenn Sie für ein Verzeichnis eine komplexe Zugriffssteuerungsliste setzen, wird diese nicht automatisch an untergeordnete Verzeichnisse dieses Verzeichnisses vererbt. Wenn Sie für ein Verzeichnis eine komplexe Zugriffssteuerungsliste setzen und diese an die untergeordneten Verzeichnisse dieses Verzeichnisses vererbt werden soll, müssen Sie die Zugriffssteuerungslisten-Vererbungsflags verwenden. Weitere Informationen finden Sie in Tabelle 8–3 und unter „Festlegen der Vererbung von Zugriffssteuerungslisten an ZFS-Dateien im ausführlichen Format“ auf Seite 259.

Erstellen einer neuen Datei und (je nach dem Wert von `umask`) einer gewöhnlichen Standard-Zugriffssteuerungsliste ähnlich wie im folgenden Beispiel:

```
$ ls -v file.1
-rw-r--r--   1 root     root      206663 May 20 14:09 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Bitte beachten Sie, dass in diesem Beispiel jede Benutzerkategorie (`owner@`, `group@`, `everyone@`) zwei Zugriffssteuerungslisteneinträge besitzt. Ein Eintrag gilt für die Verweigerung (`deny`), der andere zum Gewähren (`allow`) von Zugriffsrechten.

Es folgt eine Beschreibung dieser Datei-Zugriffssteuerungsliste:

`0:owner@` Dem Eigentümer wird das Ausführen der Datei verweigert (`execute:deny`).

`1:owner@` Der Eigentümer kann den Dateiinhalt lesen und ändern (`read_data/write_data/append_data`). Der Eigentümer kann darüber hinaus auch Dateiattribute wie z. B. Zeitmarken, erweiterte Attribute und Zugriffssteuerungslisten ändern (`write_xattr/write_attributes /write_acl`). Zusätzlich dazu kann der Eigentümer die Datei-Eigentümerschaft ändern (`write_owner:allow`).

`2:group@` Der Gruppe wird das Ausführen der Datei verweigert (`write_data/append_data/execute:deny`).

`3:group@` Der Gruppe wird die Berechtigung zum Lesen der Datei gewährt (`read_data:allow`).

`4:everyone@` Benutzern, die nicht Eigentümer der Datei sind oder nicht zur Gruppe der Dateieigentümer gehören, wird die Berechtigung zum Ausführen und Ändern des Inhalts der Datei bzw. Ändern von Dateiattributen verweigert (`write_data/append_data/write_xattr/execute/ write_attributes/write_acl/write_owner:deny`).

`5:everyone@` Benutzern, die nicht Eigentümer der Datei sind oder nicht zur Gruppe der Dateieigentümer gehören, wird die Berechtigung zum Lesen der Datei bzw. der Dateiattribute gewährt (`read_data/read_xattr/read_attributes/read_acl/ synchronize:allow`). Das Zugriffsrecht `synchronize` ist zurzeit nicht implementiert.

Erstellen eines neuen Verzeichnisses und (je nach dem Wert von `umask`) einer Standardverzeichnis-Zugriffssteuerungsliste ähnlich wie im folgenden Beispiel:

```
$ ls -dv dir.1
drwxr-xr-x   2 root     root           2 May 20 14:11 dir.1
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Es folgt eine Beschreibung dieser Verzeichnis-Zugriffssteuerungsliste:

`0:owner@` Die Verweigerungsliste für den Eigentümer ist leer (`::deny`).

`1:owner@` Der Eigentümer kann den Inhalt des Verzeichnisses lesen und ändern (`list_directory/read_data/add_file/write_data/add_subdirectory/append_data`), durchsuchen Sie die Inhalte (`execute`), und ändern Sie die Verzeichnisattribute wie z. B. Zeitmarken, erweiterte Attribute und Zugriffssteuerungslisten (`write_xattr/write_attributes/write_acl`). Zusätzlich dazu kann der Eigentümer die Verzeichnis-Eigentümerschaft ändern (`write_owner:allow`).

`2:group@` Die Gruppe kann den Verzeichnisinhalt weder erweitern noch ändern (`add_file/write_data/add_subdirectory/append_data :deny`).

`3:group@` Die Gruppe kann den Verzeichnisinhalt auflisten und lesen. Darüber hinaus kann die Gruppe kann den Verzeichnisinhalt auch durchsuchen (`list_directory/read_data/execute:allow`).

`4:everyone@` Benutzern, die nicht Eigentümer der Datei sind oder nicht zur Gruppe der Dateieigentümer gehören, wird die Berechtigung zum Erweitern bzw. Ändern des Verzeichnisinhalts verweigert (`add_file/write_data/add_subdirectory/append_data`). Außerdem wird die Berechtigung zum Ändern von Verzeichnisattributen verweigert (`write_xattr/write_attributes/write_acl/write_owner:deny`).

`5:everyone@` Benutzern, die nicht Eigentümer der Datei sind oder nicht zur Gruppe der Dateieigentümer gehören, wird die Berechtigung zum Lesen und Ausführen des Verzeichnisinhalts und der Verzeichnisattribute gewährt (`read_data/read_xattr/read_attributes/read_acl/attributes/read_acl/synchronize:allow`). Das Zugriffsrecht `synchronize` ist zurzeit nicht implementiert.

Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im ausführlichen Format

Mit dem Befehl `chmod` können Sie Zugriffssteuerungslisten von ZFS-Dateien ändern. Die folgende `chmod`-Syntax zum Ändern von Zugriffssteuerungslisten nutzt zur Erkennung des Zugriffssteuerungslistenformats die *Zugriffssteuerungslisten-Spezifikation*. Eine Beschreibung der *Zugriffssteuerungslisten-Spezifikation* finden Sie unter „Syntaxbeschreibungen zum Setzen von Zugriffssteuerungslisten“ auf Seite 245.

- Hinzufügen von Zugriffssteuerungslisten-Eintragstypen

  - Hinzufügen eines Zugriffssteuerungslisten-Eintrags für einen Benutzer

    `% chmod A+`*acl-specification filename*
  - Hinzufügen eines Zugriffssteuerungslisten-Eintrags nach *Index-ID*

    `% chmod A`*index-ID*`+`*acl-specification filename*

    Mit dieser Syntax wird der neue Zugriffssteuerungslisten-Eintrag an der entspechenden *Index-ID* eingefügt.
- Ersetzen eines Zugriffssteuerungslisten-Eintrags

  `% chmod A=`*acl-specification filename*

  `% chmod A`*index-ID*`=`*acl-specification filename*
- Entfernen von Zugriffssteuerungslisten-Einträgen
  - Entfernen eines Zugriffssteuerungslisten-Eintrags nach *Index-ID*

    `% chmod A`*index-ID*`-` *filename*
  - Entfernen eines Zugriffssteuerungslisten-Eintrags nach Benutzername

    `% chmod A-`*acl-specification filename*
  - Entfernen aller komplexen Zugriffssteuerungslisten-Einträge aus einer Datei

    `% chmod A-` *filename*

Ausführliche Informationen zu Zugriffssteuerungslisten werden mithilfe des Befehls `ls -v` angezeigt. Beispiel:

```
# ls -v file.1
-rw-r--r--   1 root     root      206663 May 20 14:09 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Informationen zur Verwendung des Zugriffssteuerungslisten-Kompaktformats finden Sie unter „Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im Kompaktformat“ auf Seite 266.

BEISPIEL 8–1 Ändern gewöhnlicher Zugriffssteuerungslisten an ZFS-Dateien

Dieser Abschnitt enthält Beispiele zum Setzen und Anzeigen gewöhnlicher Zugriffssteuerungslisten.

Im folgenden Beispiel besitzt `Datei.1` eine gewöhnliche Zugriffssteuerungsliste:

BEISPIEL 8–1 Ändern gewöhnlicher Zugriffssteuerungslisten an ZFS-Dateien *(Fortsetzung)*

```
# ls -v file.1
-rw-r--r--   1 root     root      206663 May 20 15:03 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Im folgenden Beispiel wird die Berechtigung `write_data` für `group@` gewährt:

```
# chmod A2=group@:append_data/execute:deny file.1
# chmod A3=group@:read_data/write_data:allow file.1
# ls -v file.1
-rw-rw-r--   1 root     root      206663 May 20 15:03 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:append_data/execute:deny
     3:group@:read_data/write_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Im folgenden Beispiel werden die Berechtigungen an `Datei.1 auf 644 zurückgesetzt`.

```
# chmod 644 file.1
# ls -v file.1
-rw-r--r--   1 root     root      206663 May 20 15:03 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

BEISPIEL 8–2 Setzen komplexer Zugriffssteuerungslisten an ZFS-Dateien

Dieser Abschnitt enthält Beispiele zum Setzen und Anzeigen komplexer Zugriffssteuerungslisten.

Im folgenden Beispiel werden die Berechtigungen `read_data/execute` für den Benutzer `gozer` und das Verzeichnis `test.dir` gesetzt:

BEISPIEL 8–2 Setzen komplexer Zugriffssteuerungslisten an ZFS-Dateien *(Fortsetzung)*

```
# chmod A+user:gozer:read_data/execute:allow test.dir
# ls -dv test.dir
drwxr-xr-x+  2 root     root           2 May 20 15:09 test.dir
     0:user:gozer:list_directory/read_data/execute:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Im folgenden Beispiel werden die Berechtigungen `read_data/execute` für den Benutzer `gozer` entfernt:

```
# chmod A0- test.dir
# ls -dv test.dir
drwxr-xr-x   2 root     root           2 May 20 15:09 test.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

BEISPIEL 8–3 Interaktion von Zugriffssteuerungslisten mit Berechtigungen an ZFS-Dateien

Diese Beispiele demonstrieren die Interaktion zwischen dem Setzen von Zugriffssteuerungslisten und dem anschließenden Ändern der Berechtigungen einer Datei bzw. eines Verzeichnisses.

Im folgenden Beispiel besitzt `Datei.2` eine gewöhnliche Zugriffssteuerungsliste:

```
# ls -v file.2
-rw-r--r--   1 root     root        3103 May 20 15:23 file.2
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

BEISPIEL 8–3 Interaktion von Zugriffssteuerungslisten mit Berechtigungen an ZFS-Dateien *(Fortsetzung)*

Im folgenden Beispiel werden gewährte Zugriffssteuerungslisten-Berechtigungen vom Benutzer `everyone@` entfernt:

```
# chmod A5- file.2
# ls -v file.2
-rw-r-----+  1 root     root        3103 May 20 15:23 file.2
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
```

`In dieser Ausgabe werden die Dateiberechtigungen von 644 auf 640 zurückgesetzt.` Leseberechtigungen für `everyone@` werden faktisch von den Dateiberechtigungen entfernt, wenn die gewährten Zugriffssteuerungslisten-Zugriffsrechte für `everyone@` entfernt werden.

Im folgenden Beispiel wird die vorhandene Zugriffssteuerungsliste mit den Zugriffsrechten `read_data/write_data` für `everyone@` ersetzt:

```
# chmod A=everyone@:read_data/write_data:allow file.3
# ls -v file.3
-rw-rw-rw-+  1 root     root        6986 May 20 15:25 file.3
     0:everyone@:read_data/write_data:allow
```

In dieser Ausgabe ersetzt die `chmod`-Syntax faktisch die Zugriffssteuerungsliste mit den Berechtigungen `read_data/write_data:allow` durch Lese- und Schreibberechtigungen für Eigentümer, Gruppen und `everyone@` . In diesem Modell gibt `everyone@` den Zugriff für beliebige Benutzer bzw. Gruppen an. Da keine Zugriffssteuerungslisteneinträge für `owner@` bzw. `group@ vorhanden sind, die die Zugriffsrechte für Eigentümer und Gruppen überschreiben könnten, werden die Berechtigungen auf 666 gesetzt.`

Im folgenden Beispiel wird die vorhandene Zugriffssteuerungsliste mit Leseberechtigungen für den Benutzer `gozer` ersetzt:

```
# chmod A=user:gozer:read_data:allow file.3
# ls -v file.3
----------+  1 root     root        6986 May 20 15:25 file.3
     0:user:gozer:read_data:allow
```

In diesem Beispiel werden die Dateizugriffsrechte mit `000` berechnet, da für `owner@`, `group@` bzw. `everyone@` keine Zugriffssteuerungslisteneinträge vorhanden sind, die die herkömmlichen Berechtigungskomponenten einer Datei repräsentieren. Der Dateieigentümer kann dieses Problem durch Zurücksetzen der Zugriffsrechte (und der Zugriffssteuerungsliste) wie folgt beheben:

BEISPIEL 8–3 Interaktion von Zugriffssteuerungslisten mit Berechtigungen an ZFS-Dateien *(Fortsetzung)*

```
# chmod 655 file.3
# ls -v file.3
-rw-r-xr-x+  1 root     root        6986 May 20 15:25 file.3
     0:user:gozer::deny
     1:user:gozer:read_data:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     4:group@:write_data/append_data:deny
     5:group@:read_data/execute:allow
     6:everyone@:write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/execute/read_attributes/read_acl
         /synchronize:allow
```

BEISPIEL 8–4 Wiederherstellen gewöhnlicher Zugriffssteuerungslisten an ZFS-Dateien

Mit dem Befehl `chmod` können Sie alle komplexen Zugriffssteuerungslisten einer Datei bzw. eines Verzeichnisses entfernen und dadurch die gewöhnlichen Zugriffssteuerungslisten einer Datei oder eines Verzeichnisses wiederherstellen.

Im folgenden Beispiel besitzt `test5.dir` zwei komplexe Zugriffssteuerungseinträge:

```
# ls -dv test5.dir
drwxr-xr-x+  2 root     root           2 May 20 15:32 test5.dir
     0:user:lp:read_data:file_inherit:deny
     1:user:gozer:read_data:file_inherit:deny
     2:owner@::deny
     3:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     4:group@:add_file/write_data/add_subdirectory/append_data:deny
     5:group@:list_directory/read_data/execute:allow
     6:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     7:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Im folgenden Beispiel werden die komplexen Zugriffsteuerungslisten für die Benutzer `gozer` und `lp` entfernt. Die verbleibende Zugriffsteuerungsliste enthält de sechs Standardwerte für `owner@`, `group@` und `everyone@`.

```
# chmod A- test5.dir
# ls -dv test5.dir
drwxr-xr-x   2 root     root           2 May 20 15:32 test5.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
```

BEISPIEL 8–4 Wiederherstellen gewöhnlicher Zugriffssteuerungslisten an ZFS-Dateien *(Fortsetzung)*

```
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Festlegen der Vererbung von Zugriffssteuerungslisten an ZFS-Dateien im ausführlichen Format

Sie können festlegen, ob und wie Zugriffsteuerungslisten von Dateien und Verzeichnissen vererbt werden sollen. Standardmäßig werden Zugriffssteuerungslisten nicht weitergegeben. Wenn Sie für ein Verzeichnis eine komplexe Zugriffssteuerungsliste setzen, wird diese nicht an untergeordnete Verzeichnisse vererbt. Sie müssen die Vererbung einer Zugriffssteuerungsliste für Dateien oder Verzeichnisse explizit angeben.

Darüber hinaus können zwei Eigenschaften von Zugriffssteuerungslisten in Dateisystemen global gesetzt werden: `aclinherit` und `aclmode`. Standardmäßig ist `aclinherit` auf `restricted` und `aclmode` auf `groupmask` gesetzt .

Weitere Informationen dazu finden Sie in „Vererbung von Zugriffssteuerungslisten“ auf Seite 248.

BEISPIEL 8–5 Gewähren der Standardvererbung von Zugriffssteuerungslisten

Standardmäßig werden Zugriffssteuerungslisten nicht durch eine Verzeichnisstruktur weitergegeben.

Im folgenden Beispiel wird eine komplexe Zugriffssteuerungsliste mit den Zugriffsrechten `read_data/write_data/execute` für den Benutzer `gozer` des Verzeichnisses `test.dir` gesetzt:

```
# chmod A+user:gozer:read_data/write_data/execute:allow test.dir
# ls -dv test.dir
drwxr-xr-x+  2 root     root           2 May 20 15:41 test.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Beim Erstellen eines Unterverzeichnisses in `test.dir` wird die Zugriffssteuerungsliste für den Benutzer `gozer` nicht weitergegeben. Der Benutzer `gozer` hätte nur dann Zugriff auf das

BEISPIEL 8–5 Gewähren der Standardvererbung von Zugriffssteuerungslisten *(Fortsetzung)*

Unterverzeichnis, wenn ihm die Berechtigungen für das Unterverzeichnis Zugriff als Dateieigentümer, Gruppenmitglied oder `everyone@` gewähren würden. Beispiel:

```
# mkdir test.dir/sub.dir
# ls -dv test.dir/sub.dir
drwxr-xr-x   2 root     root           2 May 20 15:42 test.dir/sub.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen

In den folgenden Beispielen sind die Zugriffssteuerungslisten für Dateien und Verzeichnisse aufgeführt, die beim Setzen des Flags `file_inherit` angewendet werden.

Im folgenden Beispiel werden die Zugriffsrechte `read_data/write_data` für den Benutzer `gozer` des Verzeichnisses `test2.dir` hinzugefügt, sodass dieser Benutzer Leseberechtigung für neu erstellte Dateien besitzt:

```
# chmod A+user:gozer:read_data/write_data:file_inherit:allow test2.dir
# ls -dv test2.dir
drwxr-xr-x+  2 root     root           2 May 20 15:50 test2.dir
     0:user:gozer:read_data/write_data:file_inherit:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Im folgenden Beispiel werden die Zugriffsrechte des Benutzers `gozer` auf die neu erstellte Datei `test2.dir/file.2` angewendet. Durch die gewährte Zugriffssteuerungslisten-Vererbung `read_data:file_inherit:allow` hat der Benutzer `gozer` Leseberechtigung für den Inhalt neu erstellter Dateien.

```
# touch test2.dir/file.2
# ls -v test2.dir/file.2
-rw-r--r--+  1 root     root           0 May 20 15:51 test2.dir/file.2
     0:user:gozer:write_data:deny
```

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen *(Fortsetzung)*

```
     1:user:gozer:read_data/write_data:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     4:group@:write_data/append_data/execute:deny
     5:group@:read_data:allow
     6:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Da die Eigenschaft `aclmode` für diese Datei auf den Standardwert `groupmask` gesetzt ist, hat der Benutzer `gozer` für die Datei `file.2` nicht das Zugriffsrecht `write_data`, da es die Gruppenberechtigung der Datei nicht zulässt.

Das Zugriffsrecht `inherit_only`, das beim Setzen des Flags `file_inherit` oder `dir_inherit` angewendet wird, dient zum Weitergeben der Zugriffssteuerungsliste durch die Verzeichnisstruktur. Somit werden dem Benutzer `gozer` nur Zugriffsrechte für `everyone@` gewährt bzw. verweigert, wenn er nicht Eigentümer der betreffenden Datei ist bzw. nicht zur Eigentümergruppe gehört. Beispiel:

```
# mkdir test2.dir/subdir.2
# ls -dv test2.dir/subdir.2
drwxr-xr-x+  2 root     root           2 May 20 15:52 test2.dir/subdir.2
     0:user:gozer:list_directory/read_data/add_file/write_data:file_inherit
         /inherit_only:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

In den folgenden Beispielen sind die Zugriffssteuerungslisten für Dateien und Verzeichnisse aufgeführt, die beim Setzen des Flags `file_inherit` und `dir_inherit` angewendet werden.

Im folgenden Beispiel werden dem Benutzer `gozer` Lese-, Schreib- und Ausführungsberechtigungen gewährt, die für neu erstellte Dateien und Verzeichnisse vererbt wurden:

```
# chmod A+user:gozer:read_data/write_data/execute:file_inherit/dir_inherit:allow
test3.dir
# ls -dv test3.dir
drwxr-xr-x+  2 root     root           2 May 20 15:53 test3.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
```

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen *(Fortsetzung)*

```
            :file_inherit/dir_inherit:allow
        1:owner@::deny
        2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
            /append_data/write_xattr/execute/write_attributes/write_acl
            /write_owner:allow
        3:group@:add_file/write_data/add_subdirectory/append_data:deny
        4:group@:list_directory/read_data/execute:allow
        5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
            /write_attributes/write_acl/write_owner:deny
        6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
            /read_acl/synchronize:allow

# touch test3.dir/file.3
# ls -v test3.dir/file.3
-rw-r--r--+  1 root     root           0 May 20 15:58 test3.dir/file.3
        0:user:gozer:write_data/execute:deny
        1:user:gozer:read_data/write_data/execute:allow
        2:owner@:execute:deny
        3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
            /write_acl/write_owner:allow
        4:group@:write_data/append_data/execute:deny
        5:group@:read_data:allow
        6:everyone@:write_data/append_data/write_xattr/execute/write_attributes
            /write_acl/write_owner:deny
        7:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
            :allow

# mkdir test3.dir/subdir.1
# ls -dv test3.dir/subdir.1
drwxr-xr-x+  2 root     root           2 May 20 15:59 test3.dir/subdir.1
        0:user:gozer:list_directory/read_data/add_file/write_data/execute
            :file_inherit/dir_inherit/inherit_only:allow
        1:user:gozer:add_file/write_data:deny
        2:user:gozer:list_directory/read_data/add_file/write_data/execute:allow
        3:owner@::deny
        4:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
            /append_data/write_xattr/execute/write_attributes/write_acl
            /write_owner:allow
        5:group@:add_file/write_data/add_subdirectory/append_data:deny
        6:group@:list_directory/read_data/execute:allow
        7:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
            /write_attributes/write_acl/write_owner:deny
        8:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
            /read_acl/synchronize:allow
```

Da die Zugriffsrechte des übergeordneten Verzeichnisses für `group@` und `everyone@` Schreib- und Ausführungsberechtigungen verweigern, wird in diesem Beispiel dem Benutzer `gozer` ebenfalls die Schreib- und Ausführungsberechtigung verweigert. Der Standardwert der Eigenschaft `aclinherit` ist `restricted`, was bedeutet, dass die Zugriffsrechte `write_data` und `execute` nicht vererbt werden.

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen *(Fortsetzung)*

Im folgenden Beispiel werden dem Benutzer `gozer` Lese-, Schreib- und Ausführungsberechtigungen gewährt, die für neu erstellte Dateien vererbt wurden. Diese Berechtigungen werden jedoch nicht an nachfolgende Verzeichnisinhalte weitergegeben.

```
# chmod A+user:gozer:read_data/write_data/execute:file_inherit/no_propagate:allow
test4.dir
# ls -dv test4.dir
drwxr-xr-x+  2 root     root           2 May 20 16:02 test4.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
         :file_inherit/no_propagate:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Wie das folgende Beispiel zeigt, werden beim Erstellen eines neuen Unterverzeichnisses die Zugriffsrechte `read_data/write_data/execute` für den Benutzer `gozer` nicht an das neue Verzeichnis `sub4.dir` weitergegeben:

```
mkdir test4.dir/sub4.dir
# ls -dv test4.dir/sub4.dir
drwxr-xr-x   2 root     root           2 May 20 16:03 test4.dir/sub4.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Wie das folgende Beispiel zeigt, werden die Zugriffsrechte `read_data/write_data/execute` für den Benutzer `gozer` an die neu erstellte Datei weitergegeben:

```
# touch test4.dir/file.4
# ls -v test4.dir/file.4
-rw-r--r--+  1 root     root           0 May 20 16:04 test4.dir/file.4
     0:user:gozer:write_data/execute:deny
     1:user:gozer:read_data/write_data/execute:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
```

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen *(Fortsetzung)*

```
     4:group@:write_data/append_data/execute:deny
     5:group@:read_data:allow
     6:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

BEISPIEL 8–7 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclmode`, die auf `passthrough` gesetzt ist

Wenn die Eigenschaft `aclmode` des Dateisystems `tank/cindys` auf `passthrough` gesetzt ist, erbt der Benutzer `gozer` die auf das Verzeichnis `test4.dir` angewendete Zugriffssteuerungsliste für die neu erstellte Datei `file.4`, wie das folgende Beispiel zeigt:

```
# zfs set aclmode=passthrough tank/cindys
# touch test4.dir/file.4
# ls -v test4.dir/file.4
-rw-r--r--+  1 root     root           0 May 20 16:08 test4.dir/file.4
     0:user:gozer:write_data/execute:deny
     1:user:gozer:read_data/write_data/execute:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     4:group@:write_data/append_data/execute:deny
     5:group@:read_data:allow
     6:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Diese Ausgabe zeigt, dass die für das übergeordnete Verzeichnis `test4.dir` gesetzte Zugriffssteuerungsliste `read_data/write_data/execute:allow:file_inherit/dir_inherit` an den Benutzer `gozer` weitergegeben wird.

BEISPIEL 8–8 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclmode`, die auf `discard` gesetzt ist

Wenn die Eigenschaft `aclmode` eines Dateisystems auf `discard` gesetzt ist, können Zugriffssteuerungslisten potenziell ignoriert werden, wenn sich die Zugriffsrechte eines Verzeichnisses ändern. Beispiel:

```
# zfs set aclmode=discard tank/cindys
# chmod A+user:gozer:read_data/write_data/execute:dir_inherit:allow test5.dir
# ls -dv test5.dir
drwxr-xr-x+  2 root     root           2 May 20 16:09 test5.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
         :dir_inherit:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
```

BEISPIEL 8–8 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclmode`, die auf `discard` gesetzt ist *(Fortsetzung)*

```
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Wenn Sie später die Zugriffsrechte an einem Verzeichnis einschränken, gilt die komplexe Zugriffssteuerungsliste nicht mehr. Beispiel:

```
# chmod 744 test5.dir
# ls -dv test5.dir
drwxr--r--   2 root     root           2 May 20 16:09 test5.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data/execute:deny
     3:group@:list_directory/read_data:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /execute/write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/read_attributes/read_acl
         /synchronize:allow
```

BEISPIEL 8–9 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclinherit`, die auf `noallow` gesetzt ist

Im folgenden Beispiel sind zwei komplexe Zugriffssteuerungslisten mit Dateivererbung gesetzt. Eine Zugriffssteuerungsliste gewährt die Berechtigung `read_data`, und die andere Zugriffssteuerungsliste verweigert die Berechtigung `read_data`. Dieses Beispiel zeigt auch, wie Sie mit dem gleichen Aufruf des Befehls `chmod` zwei Zugriffssteuerungseinträge angeben können.

```
# zfs set aclinherit=noallow tank/cindys
# chmod A+user:gozer:read_data:file_inherit:deny,user:lp:read_data:file_inherit:allow
test6.dir
# ls -dv test6.dir
drwxr-xr-x+  2 root     root           2 May 20 16:11 test6.dir
     0:user:gozer:read_data:file_inherit:deny
     1:user:lp:read_data:file_inherit:allow
     2:owner@::deny
     3:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     4:group@:add_file/write_data/add_subdirectory/append_data:deny
     5:group@:list_directory/read_data/execute:allow
     6:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     7:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

BEISPIEL 8–9 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclinherit`, die auf `noallow` gesetzt ist *(Fortsetzung)*

Wie das folgende Beispiel zeigt, gilt beim Erstellen einer neuen Datei die Zugriffssteuerungsliste, die das Zugriffsrecht `read_data` gewährt, nicht mehr.

```
# touch test6.dir/file.6
# ls -v test6.dir/file.6
-rw-r--r--   1 root     root           0 May 20 16:13 test6.dir/file.6
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im Kompaktformat

Sie können Zugriffsrechte an ZFS-Dateien in einem Kompaktformat setzen und anzeigen, das für die einzelnen Berechtigungen 14 eindeutige Buchstaben verwendet. Die Buchstaben zur kompakten Darstellung der Zugriffsrechte sind in Tabelle 8–2 und Tabelle 8–3 aufgeführt.

Kompaktausgaben von Zugriffssteuerungslisten für Dateien und Verzeichnisse werden mithilfe des Befehls `ls - V` angezeigt. Beispiel:

```
# ls -V file.1
-rw-r--r--   1 root     root      206663 Jun 17 10:07 file.1
            owner@:--x-----------:------:deny
            owner@:rw-p---A-W-Co-:------:allow
            group@:-wxp----------:------:deny
            group@:r-------------:------:allow
         everyone@:-wxp---A-W-Co-:------:deny
         everyone@:r-----a-R-c--s:------:allow
```

Es folgt eine Beschreibung dieser Kompaktausgabe:

`owner@` Dem Eigentümer wird das Ausführen der Datei verweigert (`x`= `execute`).

`owner@` Der Eigentümer kann den Dateiinhalt lesen und ändern (`rw`=`read_data/write_data`), (`p`= `append_data`). Der Eigentümer kann darüber hinaus auch Dateiattribute wie z. B. Zeitmarken, erweiterte Attribute und Zugriffssteuerungslisten ändern (`A`=`write_xattr` , `W`=`write_attributes` und `C`= `write_acl`). Zusätzlich dazu kann der Eigentümer die Datei-Eigentümerschaft ändern (`o`=`write_owner`).

`group@` Der Gruppe werden Änderungs- und Ausführungsberechtigung für die Datei verweigert (`write_data`, `p= append_data` und `x=execute`).

`group@` Der Gruppe wird die Berechtigung zum Lesen der Datei gewährt (`r= read_data`).

`everyone@` Benutzern, die nicht Eigentümer der Datei sind oder nicht zur Gruppe der Dateieigentümer gehören, wird die Berechtigung zum Ausführen und Ändern des Dateiinhalts sowie Ändern von Dateiattributen verweigert (`w=write_data`, `x= execute`, `p=append_data`, `A=write_xattr`, `W=write_attributes` , `C=write_acl` und `o= write_owner`).

`everyone@` Benutzer, die nicht Eigentümer der Datei bzw. der Dateiattribute sind oder nicht zur Gruppe der Eigentümer der Datei bzw. der Dateiattribute gehören (`r=read_data`, `a=append_data`, `R=read_xattr` , `c=read_acl` und `s= synchronize`). Das Zugriffsrecht `synchronize` ist zurzeit nicht implementiert.

Das Kompaktformat von Zugriffssteuerungslisten hat gegenüber dem ausführlichen Format folgende Vorteile:

- Zugriffsrechte können für den Befehl `chmod` als positionale Argumente angegeben werden.
- Der Bindestrich (-), der für die Verweigerung von Zugriffsrechten steht, kann weggelassen werden. Nur die erforderlichen Buchstaben müssen angegeben werden.
- Zugriffsrechte und Vererbungsflags werden in der gleichen Weise gesetzt.

Informationen zur Verwendung des ausführlichen Formats von Zugriffssteuerungslisten finden Sie unter „Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im ausführlichen Format“ auf Seite 253.

BEISPIEL 8–10 Setzen und Anzeigen von Zugriffssteuerungslisten im Kompaktformat

Im folgenden Beispiel ist eine gewöhnliche Zugriffssteuerungsliste für `file.1` vorhanden:

```
# ls -V file.1
-rw-r--r--   1 root     root      206663 Jun 17 10:07 file.1
            owner@:--x-----------:------:deny
            owner@:rw-p---A-W-Co-:------:allow
            group@:-wxp----------:------:deny
            group@:r-------------:------:allow
         everyone@:-wxp---A-W-Co-:------:deny
         everyone@:r-----a-R-c--s:------:allow
```

Im diesem Beispiel werden die Berechtigungen `read_data/execute` für den Benutzer `gozer` der Datei.1 hinzugefügt:

```
# chmod A+user:gozer:rx:allow file.1
# ls -V file.1
-rw-r--r--+  1 root     root      206663 Jun 17 10:07 file.1
        user:gozer:r-x-----------:------:allow
```

BEISPIEL 8–10 Setzen und Anzeigen von Zugriffssteuerungslisten im Kompaktformat *(Fortsetzung)*

```
            owner@:--x-----------:------:deny
            owner@:rw-p---A-W-Co-:-------:allow
            group@:-wxp----------:------:deny
            group@:r-------------:------:allow
         everyone@:-wxp---A-W-Co-:-------:deny
         everyone@:r-----a-R-c--s:------:allow
```

Eine andere Methode zum Hinzufügen der gleichen Zugriffsrechte für den Benutzer `gozer` besteht im Einfügen eines neuen Zugriffssteuerungslisteneintrags an einer neuen Position (z. B. Position 4). Somit werden die vorhandenen Zugriffssteuerungslisten an den Positionen 4-6 nach unten verschoben. Beispiel:

```
# chmod A4+user:gozer:rx:allow file.1
# ls -V file.1
-rw-r--r--+  1 root     root      206663 Jun 17 10:16 file.1
            owner@:--x-----------:------:deny
            owner@:rw-p---A-W-Co-:-------:allow
            group@:-wxp----------:------:deny
            group@:r-------------:------:allow
        user:gozer:r-x-----------:------:allow
         everyone@:-wxp---A-W-Co-:-------:deny
         everyone@:r-----a-R-c--s:------:allow
```

Im folgenden Beispiel werden dem Benutzer `gozer` Lese-, Schreib- und Ausführungsberechtigungen gewährt, die für neu erstellte Dateien und Verzeichnisse vererbt wurden.

```
# chmod A+user:gozer:rwx:fd:allow dir.2
# ls -dV dir.2
drwxr-xr-x+  2 root     root           2 Jun 17 10:19 dir.2
        user:gozer:rwx-----------:fd----:allow
            owner@:--------------:------:deny
            owner@:rwxp---A-W-Co-:-------:allow
            group@:-w-p----------:------:deny
            group@:r-x-----------:------:allow
         everyone@:-w-p---A-W-Co-:-------:deny
         everyone@:r-x---a-R-c--s:------:allow
```

Sie können Zugriffsrechte und Vererbungsflags auch aus der Ausgabe des Befehls `ls - V` in das Kompaktformat des Befehls `chmod` kopieren. Wenn Sie beispielsweise die Zugriffsrechte und Vererbungsflags von `dir.2` für den Benutzer `gozer` auf den Benutzer `cindys` von `dir.2` übertragen möchten, müssen Sie die entsprechenden Zugriffsrechte und Vererbungsflags (`rwx-----------:f-----:allow`) in den Befehl `chmod` kopieren:

```
# chmod A+user:cindys:rwx-----------:fd----:allow dir.2
# ls -dV dir.2
drwxr-xr-x+  2 root     root           2 Jun 17 10:19 dir.2
       user:cindys:rwx-----------:fd----:allow
        user:gozer:rwx-----------:fd----:allow
            owner@:--------------:------:deny
```

BEISPIEL 8–10 Setzen und Anzeigen von Zugriffssteuerungslisten im Kompaktformat *(Fortsetzung)*

```
            owner@:rwxp---A-W-Co-:-------:allow
            group@:-w-p----------:-------:deny
            group@:r-x-----------:-------:allow
         everyone@:-w-p---A-W-Co-:-------:deny
         everyone@:r-x---a-R-c--s:-------:allow
```

BEISPIEL 8–11 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclinherit`, die auf `passthrough` gesetzt ist

Wenn die Eigenschaft `aclinherit` des Dateisystems auf `passthrough` gesetzt ist, werden alle vererbbaren Zugriffssteuerungslisten ohne jegliche Änderung der Einträge bei der Vererbung weitergegeben. Ist diese Eigenschaft auf `passthrough` gesetzt, werden Dateien mit Berechtigungen erstellt, die von der vererbbaren Zugriffssteuerungsliste bestimmt werden. Wenn keine Zugriffsrechte für die vererbbaren Zugriffssteuerungseinträge vorhanden sind, werden Zugriffsrechte gesetzt, die mit der Forderung der Anwendung übereinstimmen.

In den folgenden Beispielen wird die Vererbung von Berechtigungen durch das Setzen der Eigenschaft `aclinherit` auf `passthrough` in kompakter Zugriffssteuerungslistensyntax veranschaulicht.

In diesem Beispiel wird eine Zugriffssteuerungsliste für das Verzeichnis `test1.dir` gesetzt, um die Vererbung zu erzwingen. Durch die Syntax wird für neu erstellte Dateien ein Zugriffssteuerungseintrag `owner@`, `group@` und `everyone@` erstellt. Neu erstellte Verzeichnisse erben die Zugriffssteuerungseinträge `@owner`, `group@` und `everyone@`. Darüber hinaus erben Verzeichnisse sechs weitere Zugriffssteuerungseinträge, die die Einträge an neu erstellte Verzeichnisse und Dateien weitergeben.

```
# zfs set aclinherit=passthrough tank/cindys
# pwd
/tank/cindys
# mkdir test1.dir

# chmod A=owner@:rwxpcCosRrWaAdD:fd:allow,group@:rwxp:fd:allow,everyone@::fd:allow
test1.dir
# ls -Vd test1.dir
drwxrwx---+  2 root     root           2 Jun 17 10:37 test1.dir
            owner@:rwxpdDaARWcCos:fd----:allow
            group@:rwxp----------:fd----:allow
         everyone@:--------------:fd----:allow
```

In diesem Beispiel erbt eine neu erstellte Datei die Zugriffssteuerungsliste, die für die Weitergabe an neu erstellte Dateien angegeben wurde.

```
# cd test1.dir
# touch file.1
# ls -V file.1
-rwxrwx---+  1 root     root           0 Jun 17 10:38 file.1
            owner@:rwxpdDaARWcCos:------:allow
```

BEISPIEL 8–11 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclinherit`, die auf `passthrough` gesetzt ist *(Fortsetzung)*

```
            group@:rwxp----------:------:allow
         everyone@:--------------:------:allow
```

In diesem Beispiel erbt ein neu erstelltes Verzeichnis sowohl die Zugriffssteuerungseinträge, die den Zugriff auf dieses Verzeichnis regeln, als auch diejenigen für die künftige Weitergabe an untergeordnete Objekte des neu erstellten Verzeichnisses.

```
# mkdir subdir.1
# ls -dV subdir.1
drwxrwx---+  2 root     root           2 Jun 17 10:39 subdir.1
            owner@:rwxpdDaARWcCos:fdi---:allow
            owner@:rwxpdDaARWcCos:------:allow
            group@:rwxp----------:fdi---:allow
            group@:rwxp----------:------:allow
         everyone@:--------------:fdi---:allow
         everyone@:--------------:------:allow
```

Die Einträge `-di--` und `f-i---` gelten für die Weitergabe der Vererbung und werden bei der Zugriffssteuerung nicht berücksichtigt. In diesem Beispiel wird eine Datei mit einer gewöhnlichen Zugriffssteuerungsliste in einem anderen Verzeichnis erstellt, wo keine vererbten Zugriffssteuerungseinträge vorhanden sind.

```
# cd /tank/cindys
# mkdir test2.dir
# cd test2.dir
# touch file.2
# ls -V file.2
-rw-r--r--   1 root     root           0 Jun 17 10:40 file.2
            owner@:--x-----------:------:deny
            owner@:rw-p---A-W-Co-:------:allow
            group@:-wxp----------:------:deny
            group@:r-------------:------:allow
         everyone@:-wxp---A-W-Co-:------:deny
         everyone@:r-----a-R-c--s:------:allow
```

BEISPIEL 8–12 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclinherit`, die auf `passthrough-x` gesetzt ist

Ist die Eigenschaft `aclinherit` auf `passthrough-x` gesetzt, werden Dateien mit der Ausführungsberechtigung (`x`) für `owner@`, `group@` oder `everyone@` erstellt, allerdings nur, wenn die Ausführungsberechtigung im Dateierstellungsmodus und in einem vererbbaren Zugriffssteuerungseintrag, der den Modus betrifft, eingestellt ist.

Das folgende Beispiel zeigt, wie die Ausführungsberechtigung durch Setzen der Eigenschaft `aclinherit` auf `passthrough-x` vererbt wird.

```
# zfs set aclinherit=passthrough-x tank/cindys
```

BEISPIEL 8–12 Vererbung der Zugriffssteuerungsliste mit der Eigenschaft `aclinherit`, die auf `passthrough-x` gesetzt ist *(Fortsetzung)*

Die folgende Zugriffssteuerungsliste ist auf `/tank/cindys/test1.dir` gesetzt, um ausführbare Zugriffssteuerungslisten-Vererbung für Dateien für `owner@`, `group@` und `everyone@` bereitzustellen.

```
# chmod A=owner@:rwxpcCosRrWaAdD:fd:allow,group@:rwxp:fd:allow,everyone@::fd:allow test1.dir
# ls -Vd test1.dir
drwxrwx---+  2 root     root           2 Jun 17 10:41 test1.dir
            owner@:rwxpdDaARWcCos:fd----:allow
            group@:rwxp----------:fd----:allow
         everyone@:--------------:fd----:allow
```

Es wird eine Datei (`file1`) mit den angeforderten Zugriffsrechten `0666` erstellt. Die resultierenden Zugriffsrechte sind `0660`. Die Ausführungsberechtigung wurde nicht vererbt, weil der Erstellungsmodus dies nicht fordert.

```
# touch test1.dir/file1
# ls -V test1.dir/file1
-rw-rw----+  1 root     root           0 Jun 17 10:42 test1.dir/file1
            owner@:rw-pdDaARWcCos:------:allow
            group@:rw-p----------:------:allow
         everyone@:--------------:------:allow
```

Als Nächstes wird das Executable `t` unter Verwendung des Compilers `cc` im Verzeichnis `testdir` erstellt.

```
# cc -o t t.c
# ls -V t
-rwxrwx---+  1 root     root        7396 Jun 17 10:50 t
                 owner@:rwxpdDaARWcCos:------:allow
                 group@:rwxp----------:------:allow
              everyone@:--------------:------:allow
```

Die resultierenden Zugriffsrechte sind `0770`, weil `cc` die Zugriffsrechte `0777` gefordert hat, weswegen die Ausführungsberechtigung von den Einträgen `owner@`, `group@` und `everyone@` geerbt wurde.



KAPITEL 9

Delegierte ZFS-Administration

In diesem Kapitel wird erläutert, wie mit der delegierten Administration Benutzern ohne ausreichende Zugriffsrechte ermöglicht werden kann, ZFS-Administrationsaufgaben zu erledigen.

Delegierte ZFS-Administration im Überblick

Dieses Leistungsmerkmal dient zum Verteilen fein abgestimmter Zugriffsrechte an bestimmte Benutzer, Gruppen oder an „everyone". Es werden zwei Arten der delegierten Zugriffsrechte unterstützt:

- Einzelne Zugriffsrechte wie create, destroy, mount, snapshot usw. können ausdrücklich angegeben werden.
- Auch können Gruppen von Zugriffsrechten, so genannte *Zugriffsrechtsätze*, definiert werden. Ein Zugriffsrechtsatz kann zu einem späteren Zeitpunkt aktualisiert werden. Die daraus resultierenden Änderungen wirken sich automatisch auf alle Benutzer des Satzes aus. Zugriffsrechtsätze beginnen mit dem Zeichen `@` und sind auf eine Länge von 64 Zeichen begrenzt. Für die auf das Zeichen `@` folgenden übrigen Zeichen im Namen gelten dieselben Einschränkungen wie für normale ZFS-Dateisystemnamen.

Die delegierte ZFS-Administration bietet ähnliche Möglichkeiten wie das RBAC-Sicherheitsmodell. Diese Funktion bietet für die Verwaltung von ZFS-Speicher-Pools und -Dateisystemen folgende Vorteile:

- Bei einer Migration des ZFS-Speicher-Pools werden seine Zugriffsberechtigungen mit ihm weitergegeben.

- Durch dynamische Vererbung kann gesteuert werden, wie die Zugriffsrechte durch die Dateisysteme weitergegeben werden.
- Möglich ist eine Konfiguration, bei der nur der Ersteller eines Dateisystems dieses wieder löschen kann.
- Zugriffsrechte können gezielt an bestimmte Dateisysteme verteilt werden. Neu erstellte Dateisysteme können Zugriffsrechte automatisch übernehmen.
- Es wird eine einfache NFS-Administration ermöglicht. So könnte beispielsweise ein Benutzer mit ausdrücklichen Zugriffsrechten über NFS einen Snapshot im entsprechenden `.zfs/snapshot` -Verzeichnis erstellen.

Die delegierte Administration bietet sich für die Verteilung von ZFS-Aufgaben an. Informationen zum Einsatz von RBAC für allgemeine Solaris-Administrationsaufgaben finden Sie in Teil III, „Roles, Rights Profiles, and Privileges" in *System Administration Guide: Security Services*.

Deaktivieren von delegierten ZFS-Zugriffsrechten

Die Funktionen der delegierten Administration können durch Setzen der Pool-Eigenschaft `delegation` gesteuert werden. Beispiel:

```
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  on          default
# zpool set delegation=off users
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  off         local
```

Standardmäßig ist die Eigenschaft `delegation` aktiviert.

Delegieren von ZFS-Zugriffsrechten

Mit dem Befehl `zfs allow` können Sie Nicht-Root-Benutzern Zugriffsrechte für ZFS-Datasets gewähren. Dafür stehen folgende Möglichkeiten zur Verfügung:

- Einzelne Zugriffsrechte können einem Benutzer, einer Gruppe oder global gewährt werden.
- Gruppen von Einzelzugriffsrechten können in Form von *Zugriffsrechtsätzen* ebenfalls an Benutzer, Gruppen oder global vergeben werden.
- Die Gewährung von Zugriffsrechten ist sowohl lokal für das jeweilige Dataset als auch allgemein an alle untergeordneten Objekte des aktuellen Datasets möglich.

In der folgenden Tabelle finden Sie eine Beschreibung der delegierbaren Vorgänge und aller abhängigen Zugriffsrechte, die zum Durchführen des delegierten Vorgangs erforderlich sind.

Zugriffsrecht (Unterbefehl)	Beschreibung	Abhängigkeiten
`allow`	Die Fähigkeit, eigene Zugriffsrechte an andere Benutzer weiterzugeben.	Das Zugriffsrecht, das gewährt werden soll, muss ebenfalls vorhanden sein.
`clone`	Die Fähigkeit, beliebige Snapshots des Datasets zu klonen.	Die Fähigkeiten `create` und `mount` müssen im ursprünglichen Dateisystem vorhanden sein.
`create`	Die Fähigkeit, untergeordnete Datasets zu erstellen.	Die Fähigkeit `mount` muss ebenfalls vorhanden sein.
`destroy`	Die Fähigkeit, ein Dataset zu löschen.	Die Fähigkeit `mount` muss ebenfalls vorhanden sein.
`Einhängen`	Die Fähigkeit, ein Dataset ein- und auszuhängen und Geräteverknüpfungen für Volumes zu erstellen oder zu löschen.	
`promote`	Die Fähigkeit, einen Klon zu einem Dataset zu machen.	Die Fähigkeiten `mount` und `promote` müssen im ursprünglichen Dateisystem vorhanden sein.
`receive`	Die Fähigkeit, mit dem Befehl `zfs receive` untergeordnete Dateisysteme zu erstellen.	Die Fähigkeiten `mount` und `create` müssen ebenfalls vorhanden sein.
`rename`	Die Fähigkeit, ein Dataset umzubenennen.	Die Fähigkeiten `create` und `mount` müssen im neuen übergeordneten Objekt vorhanden sein.
`rollback`	Die Fähigkeit, frühere Zustände aus einem Snapshot wiederherzustellen.	
`send`	Die Fähigkeit, einen Snapshot-Datenstrom zu senden.	
`share`	Die Fähigkeit, ein Dataset freizugeben und zu sperren.	
`snapshot`	Die Fähigkeit, Snapshots von Datasets herzustellen.	

Sie können die folgenden Zugriffsrechte erteilen, wobei diese auf Zugriff, Lesezugriff oder Bearbeitungszugriff beschränkt sein können:

- `groupquota`
- `groupused`
- `userprop`
- `userquota`
- `userused`

Darüber hinaus können Nicht-Root-Benutzern folgende ZFS-Eigenschaften delegiert werden.

- `aclinherit`
- `aclmode`
- `atime`
- `canmount`
- `casesensitivity`
- `checksum`
- `compression`
- `copies`
- `devices`
- `exec`
- `mountpoint`
- `nbmand`
- `normalization`
- `primarycache`
- `quota`
- `readonly`
- `recordsize`
- `refreservation`
- `reservation`
- `secondarycache`
- `setuid`
- `shareiscsi`
- `sharenfs`
- `sharesmb`
- `snapdir`
- `utf8only`
- `version`
- `volblocksize`
- `volsize`
- `vscan`
- `xattr`
- `zoned`

Einige dieser Eigenschaften können nur bei der Erstellung des Datasets gesetzt werden. Eine Beschreibung dieser Eigenschaften finden Sie unter „ZFS-Eigenschaften" auf Seite 189.

Delegieren von ZFS-Zugriffsrechten (`zfs allow`)

Die Syntax für `zfs allow` lautet:

```
zfs allow -[ldugecs] everyone|user|group[,...] perm|@setname,...] filesystem| volume
```

Die folgende `zfs allow`-Syntax (fett gedruckt) gibt an, wem die Zugriffsrechte übertragen werden:

`zfs allow [-uge]|user|group|everyone` `[,...]` *filesystem* | *volume*

Mehrere Entitäten können durch Komma getrennt als Liste angegeben werden. Wenn keine `-uge`-Option angegeben ist, werden die Argumente der Reihenfolge nach als das Schlüsselwort `everyone`, dann als Benutzername und schließlich als Gruppenname interpretiert. Um einen Benutzer oder eine Gruppe mit dem Namen „everyone "anzugeben, verwenden Sie die Option `-u` bzw. `-g`. Mit der Option `-g` geben Sie eine Gruppe mit demselben Namen eines Benutzers an. Die Option `-c` gewährt „Create-time"-Zugriffsrechte.

Die folgende `zfs allow`-Syntax (fett gedruckt) veranschaulicht, wie Zugriffsrechte und Zugriffsrechtsätze angegeben werden:

`zfs allow [-s] ... perm|@setname [,...]` *filesystem* | *volume*

Mehrere Zugriffsrechte können durch Komma getrennt als Liste angegeben werden. Die Namen der Zugriffsrechte sind mit den ZFS-Unterbefehlen und -Eigenschaften identisch. Weitere Informationen finden Sie im vorherigen Abschnitt.

Mehrere Zugriffsrechte lassen sich in *Zugriffsrechtsätzen* zusammenfassen. Sie werden durch die Option `-s` gekennzeichnet. Zugriffsrechtsätze können von anderen `zfs allow`-Befehlen für das angegebene Dateisystem und dessen untergeordnete Objekte verwendet werden. Zugriffsrechtsätze werden dynamisch ausgewertet, sodass Änderungen an einem Satz unverzüglich aktualisiert werden. Für Zugriffsrechtsätze gelten dieselben Benennungsregeln wie für ZFS-Dateisysteme, wobei der Name allerdings mit dem Zeichen `@` beginnen muss und nicht mehr als 64 Zeichen lang sein darf.

Die folgende `zfs allow`-Syntax (fett gedruckt) gibt an, wie die Zugriffsrechte übertragen werden:

`zfs allow [-ld]` `... ...` *filesystem* | *volume*

Die Option `-l` gibt an, dass das Zugriffsrecht für das angegebene Dataset, nicht aber für dessen untergeordnete Objekte gewährt wird, es sei denn, es wurde auch die Option `-d` angegeben. Die Option `-d` bedeutet, dass das Zugriffsrecht nicht für dieses Dataset, sondern für dessen untergeordnete Objekte gewährt wird, es sei denn, es wurde auch die Option `-l` angegeben. Wenn keine der Optionen `-ld` angegeben wird, gelten die Zugriffsrechte für das Dateisystem bzw. Volume und alle untergeordneten Objekte.

Löschen von delegierten ZFS-Zugriffsrechten (`zfs unallow`)

Mit dem Befehl `zfs unallow` können Sie zuvor erteilte Zugriffsrechte wieder löschen.

Gehen wir beispielsweise davon aus, dass Sie die Zugriffsrechte `create`, `destroy`, `mount` und `snapshot` wie folgt delegiert haben:

```
# zfs allow cindys create,destroy,mount,snapshot tank/cindys
# zfs allow tank/cindys
-------------------------------------------------------------
Local+Descendent permissions on (tank/cindys)
        user cindys create,destroy,mount,snapshot
-------------------------------------------------------------
```

Zum Entziehen dieser Zugriffsrechte müssen Sie die folgende Syntax verwenden:

```
# zfs unallow cindys tank/cindys
# zfs allow tank/cindys
```

Arbeiten mit der delegierten ZFS-Administration

In diesem Abschnitt finden Sie Beispiele zum Delegieren und Anzeigen von ZFS-Zugriffsrechten.

Delegieren von ZFS-Zugriffsrechten (Beispiele)

BEISPIEL 9–1 Delegieren von Zugriffsrechten an einzelne Benutzer

Wenn Sie einem einzelnen Benutzer die Zugriffsrechte `create` und `mount` übertragen möchten, müssen Sie sich vergewissern, dass dieser über die erforderlichen Zugriffsrechte für den zugrunde liegenden Einhängepunkt verfügt.

Um beispielsweise dem Benutzer `marks` die Zugriffsrechte `create` und `mount` für `tank` zu erteilen, setzen Sie zuerst diese Zugriffsrechte:

```
# chmod A+user:marks:add_subdirectory:fd:allow /tank
```

Anschließend gewähren Sie mit dem Befehl `zfs allow` die Zugriffsrechte `create` , `destroy` und `mount`. Beispiel:

```
# zfs allow marks create,destroy,mount tank
```

Jetzt kann der Benutzer `marks` unter `tank` eigene Dateisysteme anlegen. Beispiel:

```
# su marks
marks$ zfs create tank/marks
marks$ ^D
# su lp
$ zfs create tank/lp
cannot create 'tank/lp': permission denied
```

BEISPIEL 9–2 Delegieren der Zugriffsrechte create und destroy an Gruppen

Das folgende Beispiel veranschaulicht, wie ein Dateisystem eingerichtet wird, damit jedes Mitglied der Gruppe `staff` Dateisysteme unter `tank` erstellen und einhängen sowie eigene Dateisysteme löschen kann. Die Gruppenmitglieder von `staff` können jedoch nicht die Dateisysteme anderer Gruppenmitglieder löschen.

```
# zfs allow staff create,mount tank
# zfs allow -c create,destroy tank
# zfs allow tank
-------------------------------------------------------------
Create time permissions on (tank)
        create,destroy
Local+Descendent permissions on (tank)
        group staff create,mount
-------------------------------------------------------------
# su cindys
cindys% zfs create tank/cindys
cindys% exit
# su marks
marks% zfs create tank/marks/data
marks% exit
cindys% zfs destroy tank/marks/data
cannot destroy 'tank/mark': permission denied
```

BEISPIEL 9–3 Delegieren von Zugriffsrechten auf der richtigen Ebene der Dateisystemhierarchie

Achten Sie darauf, den Benutzern die Zugriffsrechte auf der richtigen Ebene der Dateisystemhierarchie zuweisen. So werden beispielsweise dem Benutzer `marks` die Zugriffsrechte `create`, `destroy` und `mount` für das lokale und für die untergeordneten Dateisysteme gewährt. Benutzer `marks` erhält lokale Zugriffsrechte zum Erstellen von Snapshots des Dateisystems `tank`, nicht aber des eigenen Dateisystems. Ihm wurde das Zugriffsrecht `snapshot` also nicht auf der richtigen Ebene der Dateisystemhierarchie übertragen.

```
# zfs allow -l marks snapshot tank
# zfs allow tank
-------------------------------------------------------------
Local permissions on (tank)
        user marks snapshot
Local+Descendent permissions on (tank)
        user marks create,destroy,mount
-------------------------------------------------------------
# su marks
marks$ zfs snapshot tank/@snap1
marks$ zfs snapshot tank/marks@snap1
cannot create snapshot 'mark/marks@snap1': permission denied
```

Um Benutzer `marks` Zugriffsrechte für die untergeordnete Ebene zu gewähren, verwenden Sie den Befehl `zfs allow` mit der Option `- d`. Beispiel:

```
# zfs unallow -l marks snapshot tank
# zfs allow -d marks snapshot tank
# zfs allow tank
```

BEISPIEL 9–3 Delegieren von Zugriffsrechten auf der richtigen Ebene der Dateisystemhierarchie *(Fortsetzung)*

```
-------------------------------------------------------------
Descendent permissions on (tank)
        user marks snapshot
Local+Descendent permissions on (tank)
        user marks create,destroy,mount
-------------------------------------------------------------
# su marks
$ zfs snapshot tank@snap2
cannot create snapshot 'tank@snap2': permission denied
$ zfs snapshot tank/marks@snappy
```

Jetzt kann der Benutzer `marks` nur Snapshots unterhalb der Ebene `tank` erstellen.

BEISPIEL 9–4 Definieren und Anwenden komplexer delegierter Zugriffsrechte

Benutzern oder Gruppen können spezifische Zugriffsrechte gewährt werden. So werden beispielsweise mit dem folgenden `zfs allow`-Befehl spezifische Zugriffsrechte an die Gruppe `staff` übertragen. Darüber hinaus gelten die Zugriffsrechte `destroy` und `snapshot`, sobald`tank`-Dateisysteme erstellt werden.

```
# zfs allow staff create,mount tank
# zfs allow -c destroy,snapshot tank
# zfs allow tank
-------------------------------------------------------------
Create time permissions on (tank)
        destroy,snapshot
Local+Descendent permissions on (tank)
        group staff create,mount
-------------------------------------------------------------
```

Da Benutzer `marks` ein Mitglied der Gruppe `staff` ist, kann er in `tank` Dateisysteme erstellen. Zusätzlich kann Benutzer `marks` Snapshots von `tank/marks2` erstellen, da er über die spezifischen Zugriffsrechte verfügt. Beispiel:

```
# su marks
$ zfs create tank/marks2
$ zfs allow tank/marks2
-------------------------------------------------------------
Local permissions on (tank/marks2)
        user marks destroy,snapshot
-------------------------------------------------------------
Create time permissions on (tank)
        destroy,snapshot
Local+Descendent permissions on (tank)
        group staff create
        everyone mount
-------------------------------------------------------------
```

Er kann jedoch keine Snapshots in `tank/marks` erstellen, da ihm die Zugriffsrechte hierfür fehlen. Beispiel:

BEISPIEL 9–4 Definieren und Anwenden komplexer delegierter Zugriffsrechte *(Fortsetzung)*

```
$ zfs snapshot tank/marks2@snap1
$ zfs snapshot tank/marks@snappp
cannot create snapshot 'tank/marks@snappp': permission denied
```

Wenn Sie in Ihrem home-Verzeichnis über die Berechtigung `create` verfügen, können Sie eigene Snapshot-Verzeichnisse erstellen. Dies kann sich bei Dateisystemen als hilfreich erweisen, die per NFS eingehängt werden. Beispiel:

```
$ cd /tank/marks2
$ ls
$ cd .zfs
$ ls
snapshot
$ cd snapshot
$ ls -l
total 3
drwxr-xr-x   2 marks    staff          2 Dec 15 13:53 snap1
$ pwd
/tank/marks2/.zfs/snapshot
$ mkdir snap2
$ zfs list
NAME                    USED  AVAIL  REFER  MOUNTPOINT
tank                    264K  33.2G  33.5K  /tank
tank/marks             24.5K  33.2G  24.5K  /tank/marks
tank/marks2              46K  33.2G  24.5K  /tank/marks2
tank/marks2@snap1      21.5K      -  24.5K  -
tank/marks2@snap2          0      -  24.5K  -
$ ls
snap1  snap2
$ rmdir snap2
$ ls
snap1
```

BEISPIEL 9–5 Definieren und Anwenden von delegierten Zugriffsrechtsätzen für ZFS

Das folgende Beispiel zeigt, wie der Zugriffsrechtsatz `@myset` erstellt wird und dieser und das Zugriffsrecht rename an die Gruppe `staff` für das Dateisystem `tank` übertragen werden. Der Benutzer `cindys`, Mitglied der Gruppe `staff`, verfügt über die Berechtigung zum Erstellen von Dateisystemen in `tank`. Benutzer `lp` verfügt jedoch nicht über die Berechtigung zum Erstellen von Dateisystemen in `tank`.

```
# zfs allow -s @myset create,destroy,mount,snapshot,promote,clone,readonly tank
# zfs allow tank
-------------------------------------------------------------
Permission sets on (tank)
        @myset clone,create,destroy,mount,promote,readonly,snapshot
-------------------------------------------------------------
# zfs allow staff @myset,rename tank
# zfs allow tank
-------------------------------------------------------------
Permission sets on (tank)
        @myset clone,create,destroy,mount,promote,readonly,snapshot
```

BEISPIEL 9–5 Definieren und Anwenden von delegierten Zugriffsrechtsätzen für ZFS *(Fortsetzung)*

```
Local+Descendent permissions on (tank)
        group staff @myset,rename
# chmod A+group:staff:add_subdirectory:fd:allow tank
# su cindys
cindys% zfs create tank/data
Cindys% zfs allow tank
-------------------------------------------------------------
Permission sets on (tank)
        @myset clone,create,destroy,mount,promote,readonly,snapshot
Local+Descendent permissions on (tank)
        group staff @myset,rename
-------------------------------------------------------------
cindys% ls -l /tank
total 15
drwxr-xr-x   2 cindys   staff          2 Aug  8 14:10 data
cindys% exit
# su lp
$ zfs create tank/lp
cannot create 'tank/lp': permission denied
```

Anzeigen von delegierten ZFS-Zugriffsrechten (Beispiele)

Mit dem folgenden Befehl werden die Zugriffsrechte angezeigt:

```
# zfs allow dataset
```

Dieser Befehl gibt die für das jeweilige Dataset gesetzten oder gewährten Zugriffsrechte aus. Die Ausgabe enthält folgende Komponenten:

- Zugriffsrechtsätze
- Spezifische Zugriffsrechte oder Create-time-Zugriffsrechte
- Lokales Dataset
- Lokale und untergeordnete Datasets
- Nur untergeordnete Datasets

BEISPIEL 9–6 Anzeigen einfacher Zugriffsrechte für die delegierte Administration

Die folgende Ausgabe in diesem Beispiel bedeutet, dass der Benutzer `cindys` berechtigt ist, im Dateisystem `tank/cindys` Objekte zu erstellen, zu löschen, einzuhängen und Snapshots zu erstellen.

```
# zfs allow tank/cindys
       -------------------------------------------------------------
       Local+Descendent permissions on (tank/cindys)
               user cindys create,destroy,mount,snapshot
```

BEISPIEL 9–7 Anzeigen komplexer Zugriffsrechte für die delegierte Administration

Die Ausgabe in diesem Beispiel zeigt die folgenden Zugriffsrechte für die Dateisysteme `pool/fred` und `pool` auf.

Für das Dateisystem `pool/fred`:

- Es sind zwei Zugriffsrechtsätze definiert:
  - `@eng` (create, destroy, snapshot, mount, clone, promote, rename)
  - `@simple` (create, mount)
- Die Create-time-Zugriffsrechte werden für den Zugriffsrechtsatz `@eng` und die Eigenschaft `mountpoint` gesetzt. „Create-time" bedeutet, dass der Zugriffsrechtsatz `@eng` und die Eigenschaft `mountpoint` nach der Erstellung eines Datasets gewährt werden.
- Dem Benutzer `tom` wird der Zugriffsrechtsatz `@eng` und dem Benutzer `joe` werden die Zugriffsrechte `create`, `destroy` und `mount` für lokale Dateisysteme gewährt.
- Benutzer `fred` erhält den Zugriffsrechtsatz `@basic` sowie die Zugriffsrechte `share` und `rename` für das lokale und die untergeordneten Dateisysteme.
- Dem Benutzer `barney` und der Gruppe `staff` wird der Zugriffsrechtsatz `@basic` ausschließlich für untergeordnete Dateisysteme übertragen.

Für das Dateisystem `pool`:

- Der Zugriffsrechtsatz `@simple` (create, destroy, mount) ist definiert.
- Die Gruppe `staff` erhält den Zugriffsrechtsatz `@simple` für das lokale Dateisystem.

Sehen Sie hier die Ausgabe für dieses Beispiel:

```
$ zfs allow pool/fred
------------------------------------------------------------------------------
Permission sets on (pool/fred)
        @eng create,destroy,snapshot,mount,clone,promote,rename
        @simple create,mount
Create time permissions on (pool/fred)
        @eng,mountpoint
Local permissions on (pool/fred)
        user tom @eng
        user joe create,destroy,mount
Local+Descendent permissions on (pool/fred)
        user fred @basic,share,rename
Descendent permissions on (pool/fred)
        user barney @basic
        group staff @basic
------------------------------------------------------------------------------
Permission sets on (pool)
        @simple create,destroy,mount
Local permissions on (pool)
        group staff @simple
------------------------------------------------------------------------------
```

Löschen von ZFS-Zugriffsrechten (Beispiele)

Mit dem Befehl `zfs unallow` lassen sich übertragene Zugriffsrechte wieder löschen. Beispielsweise verfügt Benutzer `cindys` über die Zugriffsrechte zum Erstellen, Löschen, Einhängen und Anfertigen von Snapshots im Dateisystem `tank/cindys`.

```
# zfs allow cindys create,destroy,mount,snapshot tank/cindys
       # zfs allow tank/cindys
       -------------------------------------------------------------
       Local+Descendent permissions on (tank/cindys)
               user cindys create,destroy,mount,snapshot
       -------------------------------------------------------------
```

Mit der folgenden `zfs unallow`-Syntax entziehen Sie das Zugriffsrecht von Benutzer `cindys` zum Erstellen von Snapshots für das Dateisystem `tank/cindys`:

```
# zfs unallow cindys snapshot tank/cindys
# zfs allow tank/cindys
-------------------------------------------------------------
Local+Descendent permissions on (tank/cindys)
        user cindys create,destroy,mount
-------------------------------------------------------------
cindys% zfs create tank/cindys/data
cindys% zfs snapshot tank/cindys@today
cannot create snapshot 'tank/cindys@today': permission denied
```

Betrachten wir als weiteres Beispiel die nachfolgenden Zugriffsrechte des Benutzers `marks` in `tank/marks`:

```
# zfs allow tank/marks
-------------------------------------------------------------
Local+Descendent permissions on (tank/marks)
        user marks create,destroy,mount
-------------------------------------------------------------
```

In diesem Beispiel werden mit der folgenden `zfs unallow`-Syntax alle Zugriffsrechte für Benutzer `marks` von `tank/marks` gelöscht:

```
# zfs unallow marks tank/marks
```

Die folgende `zfs unallow`-Syntax löscht einen Zugriffsrechtsatz für das Dateisystem `tank`.

```
# zfs allow tank
-------------------------------------------------------------
Permission sets on (tank)
        @myset clone,create,destroy,mount,promote,readonly,snapshot
Create time permissions on (tank)
        create,destroy,mount
Local+Descendent permissions on (tank)
        group staff create,mount
-------------------------------------------------------------
# zfs unallow -s @myset tank
$ zfs allow tank
```

```
-------------------------------------------------------------
Create time permissions on (tank)
        create,destroy,mount
Local+Descendent permissions on (tank)
        group staff create,mount
-------------------------------------------------------------
```



Fortgeschrittene Oracle Solaris ZFS-Themen

Dieses Kapitel enthält Informationen zu ZFS-Volumes, zur Verwendung von ZFS auf Solaris-Systemen mit installierten Zonen, zu Speicher-Pools mit alternativem Root-Verzeichnis sowie zu ZFS-Zugriffsrechtsprofilen.

Dieses Kapitel enthält die folgenden Abschnitte:

ZFS-Volumes

Ein ZFS-Volume ist ein Dataset, das eine Blockeinheit darstellt. ZFS-Volumes sind im Verzeichnis `/dev/zvol/{dsk,rdsk}/pool` als Geräte aufgeführt.

Im folgenden Beispiel wird ein ZFS-Volume `tank/vol` mit einer Kapazität von 5 GB erstellt:

```
# zfs create -V 5gb tank/vol
```

Beim Erstellen von Volumes wird automatisch eine Reservierung der anfänglichen Volume-Kapazität angelegt, um unerwartetes Verhalten zu verhindern. Wenn die Kapazität des Volumes beispielsweise kleiner wird, können Daten beschädigt werden. Deswegen müssen Sie beim Ändern der Kapazität eines Volumes äußerst sorgfältig vorgehen.

Außerdem können Dateisysteminkonsistenzen entstehen, wenn Sie einen Snapshot eines Volumes erstellen, dessen Kapazität sich ändern kann, und versuchen, den betreffenden Snapshot mittels Rollback rückgängig zu machen oder zu klonen.

Informationen zu Dateisystemeigenschaften, die auf Volumes angewendet werden können, finden Sie in Tabelle 6–1.

Auf Solaris-Systemen mit installierten Zonen können Sie keine ZFS-Volumes in einer nicht-globalen Zone erstellen bzw. klonen. Alle solche Versuche schlagen fehl. Informationen zur Verwendung von ZFS-Volumes in einer globalen Zone finden Sie unter „Hinzufügen von ZFS-Volumes zu einer nicht-globalen Zone“ auf Seite 293.

Verwendung von ZFS-Volumes als Swap- bzw. Dump-Gerät

Während der Installation eines ZFS-Root-Dateisystems oder einer Migration von einem UFS-Root-Dateisystem wird auf einem ZFS-Volume im ZFS-Root-Pool ein Swap-Gerät erstellt. Beispiel:

```
# swap -l
swapfile                  dev    swaplo   blocks     free
/dev/zvol/dsk/rpool/swap 253,3        16  8257520  8257520
```

Während der Installation eines ZFS-Root-Dateisystems oder einer Migration von einem UFS-Root-Dateisystem wird auf einem ZFS-Volume im ZFS-Root-Pool ein Dump-Gerät erstellt. Nach der Einrichtung ist keine Verwaltung des Dump-Geräts erforderlich. Beispiel:

```
# dumpadm
      Dump content: kernel pages
       Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash/t2000
  Savecore enabled: yes
```

Wenn Sie den Swap-Bereich oder das Dump-Gerät nach der Installation oder dem Upgrade des Systems ändern müssen, benutzen Sie hierzu die Befehle `swap` und `dumpadm` wie in vorherigen Solaris-Versionen. Zum Erstellen eines zusätzlichen Swap-Volumes müssen Sie ein ZFS-Volume einer bestimmten Kapazität erstellen und für dieses Gerät dann die Swap-Funktion aktivieren. Beispiel:

```
# zfs create -V 2G rpool/swap2
# swap -a /dev/zvol/dsk/rpool/swap2
# swap -l
swapfile                   dev  swaplo blocks   free
/dev/zvol/dsk/rpool/swap  256,1      16 2097136 2097136
/dev/zvol/dsk/rpool/swap2 256,5      16 4194288 4194288
```

Swaps dürfen nicht in eine Datei eines ZFS-Dateisystems durchgeführt werden. ZFS unterstützt keine Konfigurationen für Swap-Dateien.

Informationen zum Anpassen der Größe von Swap- und Dump-Volumes finden Sie unter „Anpassen der Größe von ZFS-Swap- und Dump-Geräten“ auf Seite 167.

Verwendung von ZFS-Volumes als Solaris-iSCSI-Zielgerät

Sie können ZFS-Volumes auf einfache Weise als iSCSI-Zielgerät konfigurieren, indem Sie die Eigenschaft `shareiscsi` für das Volume setzen. Beispiel:

```
# zfs create -V 2g tank/volumes/v2
# zfs set shareiscsi=on tank/volumes/v2
# iscsitadm list target
Target: tank/volumes/v2
    iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
    Connections: 0
```

Nach dem Erstellen des iSCSI-Zielgeräts muss der iSCSI-Initiator definiert werden. Weitere Informationen zu Solaris-iSCSI-Zielgeräten und -Initiatoren finden Sie in Kapitel 14, „Configuring Oracle Solaris iSCSI Targets and Initiators (Tasks)“ in *System Administration Guide: Devices and File Systems*.

Hinweis – Solaris-iSCSI-Zielgeräte können auch mit dem Befehl `iscsitadm` erstellt und verwaltet werden. Wenn die Eigenschaft `shareiscsi` eines ZFS-Volumes gesetzt ist, darf der Befehl `iscsitadm` nicht zum Erstellen des gleichen Zielgeräts verwendet werden, da ansonsten für das gleiche Zielgerät duplizierte Zielgerätinformationen erstellt werden.

Als iSCSI-Zielgerät konfigurierte ZFS-Volumes werden genauso wie andere ZFS-Datasets verwaltet. Operationen zum `Umbenennen`, `Exportieren` und `Importieren` funktionieren bei iSCSI-Zielgeräten jedoch etwas anders.

- Wenn Sie ein ZFS-Volume umbenennen, bleibt der Name des iSCSI-Zielgeräts gleich. Beispiel:

  ```
  # zfs rename tank/volumes/v2 tank/volumes/v1
  # iscsitadm list target
  Target: tank/volumes/v1
      iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
      Connections: 0
  ```
- Durch das Exportieren eines Pools, das ein für den Netzwerkzugriff freigegebenes ZFS-Volume enthält, wird das betreffende Zielgerät entfernt. Durch das Importieren eines Pools, das ein freigegebenes ZFS-Volume enthält, wird das betreffende Zielgerät für den Netzwerkzugriff freigegeben. Beispiel:

  ```
  # zpool export tank
  # iscsitadm list target
  # zpool import tank
  # iscsitadm list target
  Target: tank/volumes/v1
      iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
      Connections: 0
  ```

Alle Konfigurationsinformationen zu einem iSCSI-Zielgerät werden im Dataset gespeichert. Wie bei einem über NFS für den Netzwerkzugriff freigegebenem Dateisystem, wird ein in ein anderes System importiertes iSCSI-Zielgerät entsprechend für den Netzwerkzugriff freigegeben.

Verwendung von ZFS in einem Solaris-System mit installierten Zonen

In den folgenden Abschnitten wird die Verwendung von ZFS auf Systemen mit Oracle Solaris-Zonen beschrieben:

- „Hinzufügen von ZFS-Dateisystemen zu einer nicht-globalen Zone“ auf Seite 291
- „Delegieren von Datasets in eine nicht-globale Zone“ auf Seite 292
- „Hinzufügen von ZFS-Volumes zu einer nicht-globalen Zone“ auf Seite 293
- „Verwenden von ZFS-Speicher-Pools innerhalb einer Zone“ auf Seite 293
- „Verwalten von ZFS-Eigenschaften innerhalb einer Zone“ auf Seite 294
- „Informationen zur Eigenschaft `zoned`“ auf Seite 295

Informationen zum Konfigurieren eines Systems mit einem ZFS-Root-Dateisystem, das mithilfe von Oracle Solaris Live Upgrade migriert oder gepatcht werden soll, finden Sie unter „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 150 oder „Verwenden des Oracle Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (ab Solaris 10 5/09)“ auf Seite 156.

Für die Zusammenarbeit von ZFS-Datasets mit Zonen sollten Sie folgende Aspekte berücksichtigen:

- Sie können ein ZFS-Dateisystem bzw. einen ZFS-Klon mit oder ohne Delegierung der Verwaltungssteuerung zu einer nicht-globalen Zone hinzufügen.
- ZFS-Volumes können als Gerät zu nicht-globalen Zonen hinzugefügt werden.
- Derzeit können ZFS-Snapshots nicht in Zonen erstellt werden.

In den folgenden Abschnitten wird mit dem Begriff „ZFS-Dataset” ein Dateisystem bzw. Klon bezeichnet.

Durch Hinzufügen eines Datasets kann sich einen nicht-globale Zone Festplattenkapazität mit der globalen Zone teilen, obwohl der Zonen-Administrator in der zugrunde liegenden Dateisystemhierarchie keine Eigenschaften einstellen bzw. keine neuen Dateisysteme erstellen kann. Dieser Vorgang ist mit dem Hinzufügen anderer Dateisystemtypen zu einer Zone identisch und sollte dann verwendet werden, wenn das Hauptziel in der gemeinsamen Nutzung von Festplattenkapazität besteht.

In ZFS können Datasets auch an eine nicht-globale Zone delegiert werden. Dadurch erlangt der Zonen-Administrator vollständige Kontrolle über diese Datasets und alle seine untergeordneten Objekte. Der Zonen-Administrator kann Dateisysteme bzw. Klone in diesem

Dataset erstellen oder löschen sowie Dataset-Eigenschaften ändern. Der Zonen-Administrator hat jedoch keine Kontrolle über Datasets, die nicht zur betreffenden Zone hinzugefügt wurden, und darf die für die oberste Hierarchieebene des delegierten Datasets gesetzten Kontingente nicht überschreiten.

Bei der Verwendung von ZFS auf Systemen mit installierten Oracle Solaris-Zonen sollten Sie Folgendes berücksichtigen:

- Die Eigenschaft `mountpoint eines zu einer nicht-globalen Zone hinzugefügten ZFS-Dateisystems muss auf „legacy" gesetzt sein`.
- Sehen Sie aufgrund von CR 6449301 davon ab, einer nicht-globalen Zone bei der Konfiguration ein ZFS-Dataset hinzuzufügen. Sie können das·ZFS-Dataset hinzufügen, nachdem die Zone installiert wurde.
- Befinden sich der `zonepath` der Quelle und der `zonepath` des Ziels in einem ZFS-Dateisystem und im gleichen Pool, so verwendet der Befehl `zoneadm clone` nun automatisch die ZFS-Klonfunktion, um die Zone zu klonen. Mit dem Befehl `zoneadm clone` wird ein ZFS-Snapshot des `zonepath`-Quellverzeichnisses erstellt und das `zonepath`-Zielverzeichnis eingerichtet. Das Klonen einer Zone mit dem Befehl `zfs clone` ist nicht möglich. Weitere Informationen finden Sie in Teil II, „Zonen" in *Systemverwaltungshandbuch: Oracle Solaris Container – Ressourcenverwaltung und Solaris Zones*.
- Wenn Sie ein ZFS-Dateisystem an eine nicht-globale Zone delegieren, müssen Sie dieses Dateisystem vor der Verwendung des Oracle Solaris Live Upgrade aus der nicht-globalen Zone entfernen. Andernfalls kann das Oracle Live Upgrade nicht ausgeführt werden, was auf einen Fehler im Zusammenhang mit dem Schreibschutz des Dateisystems zurückzuführen ist.

Hinzufügen von ZFS-Dateisystemen zu einer nicht-globalen Zone

Wenn das Hauptziel ausschließlich in der gemeinsamen Nutzung von Speicherplatz mit der globalen Zone besteht, können Sie ein ZFS-Dateisystem zu einer nicht-globalen Zone als generisches Dateisystem hinzufügen. Die Eigenschaft `mountpoint eines zu einer nicht-globalen Zone hinzugefügten ZFS-Dateisystems muss auf „legacy" gesetzt sein`.

Sie können ein ZFS-Dateisystem mit dem Unterbefehl `add fs` des Befehls `zonecfg` zu einer nicht-globalen Zone hinzufügen.

Im folgenden Beispiel wird ein ZFS-Dateisystem durch den Administrator der globalen Zone aus der globalen Zone zu einer nicht-globalen Zone hinzugefügt:

```
# zonecfg -z zion
zonecfg:zion> add fs
zonecfg:zion:fs> set type=zfs
zonecfg:zion:fs> set special=tank/zone/zion
zonecfg:zion:fs> set dir=/export/shared
zonecfg:zion:fs> end
```

Diese Syntax fügt das ZFS-Dateisystem `tank/zone/zion` zur bereits konfigurierten und unter `/export/shared` eingehängten Zone `zion` hinzu. Die Eigenschaft `mountpoint` des Dateisystems muss auf `legacy` gesetzt sein, und das Dateisystem darf nicht schon irgendwo anders eingehängt sein. Der Zonenadministrator kann Dateien im Dateisystem erstellen und löschen. Das Dateisystem kann nicht an einer anderen Stelle eingehängt werden, und der Zonenadministrator kann keine Dateisystemeigenschaften wie z. B. `atime`, `readonly`, `compression` usw. ändern. Der Administrator der globalen Zone ist für das Einstellen und Steuern von Dateisystemeigenschaften verantwortlich.

Weitere Informationen zum Befehl `zonecfg` sowie zur Konfiguration von Ressourcentypen mithilfe von `zonecfg` finden Sie unter Teil II, „Zonen" in *Systemverwaltungshandbuch: Oracle Solaris Container – Ressourcenverwaltung und Solaris Zones*.

Delegieren von Datasets in eine nicht-globale Zone

Wenn das Hauptziel in der Delegierung der Speicherplatzverwaltung an eine Zone besteht, können Datasets mit dem Unterbefehl `add dataset` des Befehls `zonecfg` zu einer nicht-globalen Zone hinzugefügt werden.

Im folgenden Beispiel wird ein ZFS-Dateisystem durch den Administrator der globalen Zone aus der globalen Zone an eine nicht-globalen Zone delegiert:

```
# zonecfg -z zion
zonecfg:zion> add dataset
zonecfg:zion:dataset> set name=tank/zone/zion
zonecfg:zion:dataset> end
```

Im Gegensatz zum Hinzufügen von Dateisystemen wird durch diese Syntax das ZFS-Dateisystem `tank/zone/zion` in der bereits konfigurierten Zone `zion` sichtbar. Der Zonenadministrator kann Dateisystemeigenschaften einstellen und untergeordnete Dateisysteme erstellen. Darüber hinaus kann der Zonenadministrator Snapshots sowie Klone erstellen und die Dateisystemhierarchie auch anderweitig steuern.

Wenn Sie Oracle Solaris Live Upgrade zur Aktualisierung der ZFS-BU mit nicht-globalen Zonen verwenden, sollten Sie vorher jegliche delegierte Datasets entfernen. Andernfalls kann das Oracle Solaris Live Upgrade nicht ausgeführt werden, was auf einen Fehler im Zusammenhang mit dem Schreibschutz des Dateisystems zurückzuführen ist. Beispiel:

```
zonecfg:zion>
zonecfg:zion> remove dataset name=tank/zone/zion
zonecfg:zion1> exit
```

Weitere Informationen zu zulässigen Aktionen innerhalb von Zonen finden Sie unter „Verwalten von ZFS-Eigenschaften innerhalb einer Zone“ auf Seite 294.

Hinzufügen von ZFS-Volumes zu einer nicht-globalen Zone

ZFS-Volumes können nicht mit dem Unterbefehl `add dataset` des Befehls `zonecfg` zu einer nicht-globalen Zone hinzugefügt werden. Mithilfe des Unterbefehls `add device` des Befehls `zonecfg` können Zonen jedoch Volumes hinzugefügt werden.

Im folgenden Beispiel wird ein ZFS-Volume durch den Administrator der globalen Zone aus der globalen Zone zu einer nicht-globalen Zone hinzugefügt:

```
# zonecfg -z zion
zion: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:zion> create
zonecfg:zion> add device
zonecfg:zion:device> set match=/dev/zvol/dsk/tank/vol
zonecfg:zion:device> end
```

Mit dieser Syntax wird das Volume `tank/vol` zur Zone`zion` hinzugefügt. Bitte beachten Sie, dass das Hinzufügen eines Volumes im Raw-Modus auch dann Sicherheitsrisiken in sich birgt, wenn das Volume keinem physischen Datenspeichergerät entspricht. Der Zonenadministrator könnte dadurch insbesondere fehlerhafte Dateisysteme erstellen, die beim Einhängen einen Systemabsturz verursachen. Weitere Informationen zum Hinzufügen von Geräten zu Zonen und den damit verbundenen Sicherheitsrisiken finden Sie unter „Informationen zur Eigenschaft `zoned`“ auf Seite 295.

Weitere Informationen zum Hinzufügen von Geräten zu Zonen finden Sie unter Teil II, „Zonen“ in *Systemverwaltungshandbuch: Oracle Solaris Container – Ressourcenverwaltung und Solaris Zones*.

Verwenden von ZFS-Speicher-Pools innerhalb einer Zone

ZFS-Speicher-Pools können in Zonen werder erstellt noch geändert werden. Das Modell der delegierten Administration zentralisiert die Verwaltung physischer Datenspeichergeräte in der globalen Zone und die Verwaltung von virtuellem Speicherplatz in den nicht-globalen Zonen. Obwohl Datasets auf Pool-Ebene zu einer Zone hinzugefügt werden können, sind Befehle, die die physischen Eigenschaften eines Pools ändern (z. B. Erstellen, Hinzufügen oder Entfernen von Datenspeichergeräten) innerhalb einer Zone nicht zulässig. Auch wenn physische Datenspeichergeräte mit dem Unterbefehl `add device` des Befehls `zonecfg` zu einer Zone hinzugefügt oder Dateien verwendet werden, erlaubt der Befehl `zpool` kein Erstellen neuer Pools innerhalb der Zone.

Verwalten von ZFS-Eigenschaften innerhalb einer Zone

Nach dem Delegieren eines Datasets an eine Zone kann der Zonenadministrator bestimmte Dataset-Eigenschaften einstellen. Wenn ein Dataset an eine Zone delegiert wird, sind alle seine übergeordneten Datasets als schreibgeschützte Datasets sichtbar. Das Dataset selbst sowie alle seine untergeordneten Datasets besitzen diesen Schreibschutz jedoch nicht. Betrachten wir zum Beispiel die folgende Konfiguration:

```
global# zfs list -Ho name
tank
tank/home
tank/data
tank/data/matrix
tank/data/zion
tank/data/zion/home
```

Würde `tank/data/zion` zu einer Zone hinzugefügt, besäße jedes Dataset folgende Eigenschaften.

Dataset	Sichtbar	Beschreibbar	Unveränderbare Eigenschaften
`tank`	Ja	Nein	-
`tank/home`	Nein	-	-
`tank/data`	Ja	Nein	-
`tank/data/matrix`	Nein	-	-
`tank/data/zion`	Ja	Ja	`sharenfs`, `zoned`, `quota`, `reservation`
`tank/data/zion/home`	Ja	Ja	`sharenfs`, `zoned`

Bitte beachten Sie, dass übergeordnete Datasets von `tank/zone/zion` schreibgeschützt sichtbar, alle untergeordneten Datasets beschreibbar und nicht zur übergeordneten Hierarchie gehörende Datasets nicht sichtbar sind. Der Zonenadministrator kann die Eigenschaft `sharenfs` nicht ändern, da nicht-globale Zonen nicht als NFS-Server fungieren können. Der Zonenadministrator kann nicht die Eigenschaft `zoned` ändern, da dies ein Sicherheitsrisiko (siehe nächster Abschnitt) darstellen würde.

Privilegierte Benutzer in der Zone können jede andere einstellbare Eigenschaft ändern, ausgenommen die Eigenschaften `quota` und `reservation`. Durch dieses Verhalten kann der globale Zonenadministrator den Verbrauch von Festplattenkapazität aller Datasets in den nicht-globalen Zonen überwachen.

Außerdem können nach dem Delegieren eines Datasets an eine nicht-globalen Zone die Eigenschaften `sharenfs` und `mountpoint` vom globalen Zonenadministrator nicht geändert werden.

Informationen zur Eigenschaft `zoned`

Beim Delegieren eines Datasets an eine nicht-globale Zone muss dieses Dataset entsprechend gekennzeichnet werden, sodass bestimmte Eigenschaften nicht im Kontext der globalen Zone interpretiert werden. Nachdem ein Dataset an eine nicht-globalen Zone delegiert wurde und unter der Kontrolle eines Zonenadministrators ist, ist dessen Inhalt nicht mehr vertrauenswürdig. Wie bei anderen Dateisystemen auch können `setuid`-Binärdateien, symbolische Verknüpfungen oder andere fragwürdige Inhalte auftreten, die sich negativ auf die Sicherheit der globalen Zone auswirken. Darüber hinaus kann die Eigenschaft `mountpoint` nicht im Kontext der globalen Zone interpretiert werden, da der Zonenadministrator andernfalls den Namensplatz der globalen Zone ändern kann. Zur Lösung des letzteren Problems nutzt ZFS die Eigenschaft `zoned`, die anzeigt, dass ein Dataset zu einem bestimmten Zeitpunkt an eine nicht-globale Zone delegiert wurde.

Die Eigenschaft `zoned` ist ein boolescher Wert, der beim Booten einer Zone, die ein ZFS-Dataset enthält, automatisch auf „true" gesetzt wird. Diese Eigenschaft muss vom Zonenadministrator nicht manuell gesetzt werden. Wenn die Eigenschaft `zoned` gesetzt ist, kann das Dataset in der globalen Zone nicht eingehängt bzw. für den Netzwerkzugriff freigegeben werden. Im folgenden Beispiel wurde `tank/zone/zion` an eine Zone delegiert und `tank/zone/global` nicht:

```
# zfs list -o name,zoned,mountpoint -r tank/zone
NAME                  ZONED  MOUNTPOINT
tank/zone/global        off  /tank/zone/global
tank/zone/zion           on  /tank/zone/zion
# zfs mount
tank/zone/global            /tank/zone/global
tank/zone/zion              /export/zone/zion/root/tank/zone/zion
```

Bitte beachten Sie den Unterschied zwischen der Eigenschaft `mountpoint` und dem Verzeichnis, in das das Dataset `tank/zone/zion` gegenwärtig eingehängt ist. Die Eigenschaft `mountpoint` zeigt an, wo das Dataset auf dem Datenträger gespeichert ist, und nicht, wo es gegenwärtig eingehängt ist.

Beim Entfernen eines Datasets aus einer Zone bzw. Löschen einer Zone wird die Eigenschaft `zoned` *nicht* automatisch auf „false" zurückgesetzt. Dieses Verhalten resultiert aus den diesen Aufgaben innewohnenden Sicherheitsrisiken. Da ein nicht vertrauenswürdiger Benutzer vollständigen Zugriff auf das Dataset und seine untergeordneten Datasets hatte, kann die Eigenschaft `mountpoint` auf ungültige Werte gesetzt worden sein oder es können in den Dateisystemen `setuid`-Binärdateien vorhanden sein.

Zum Vermeiden versehentlicher Sicherheitsrisiken muss die Eigenschaft `zoned` vom Administrator der globalen Zone manuell zurückgesetzt werden, wenn das betreffende Dataset wiederverwendet werden soll. Bevor Sie die Eigenschaft `zoned` auf `off` setzen, müssen Sie sich vergewissern, dass die Eigenschaft `mountpoint` des Datasets und aller seiner untergeordneten Datasets auf zulässige Werte gesetzt ist und keine `setuid`-Binärdateien vorhanden sind, oder die Eigenschaft `setuid` deaktivieren.

Nachdem Sie sich überzeugt haben, dass keine Sicherheitslücken vorhanden sind, können Sie die Eigenschaft `zoned` mithilfe des Befehls `zfs set` oder `zfs inherit` deaktivieren. Wenn die Eigenschaft `zoned` deaktiviert wird, wenn das betreffende Dataset innerhalb einer Zone verwendet wird, kann das System unvorhersehbar reagieren. Deswegen sollten Sie den Wert dieser Eigenschaft nur dann ändern, wenn Sie sich sicher sind, dass das Dataset nicht mehr von einer nicht-globalen Zone verwendet wird.

Verwenden von ZFS-Speicher-Pools mit alternativem Root-Verzeichnis

Beim Erstellen eines Pools wird dieser inhärent mit dem Host-System verknüpft. Das Host-System verwaltet die Informationen des Pools und erkennt, wenn dieser nicht verfügbar ist. Obwohl für den Normalbetrieb nützlich, können diese Informationen hinderlich sein, wenn Sie das System von einem alternativen Datenträger booten oder ein Pool auf Wechsel-Datenträgern erstellen. Zur Lösung dieses Problems bietet ZFS *Speicher-Pools mit alternativem Root-Verzeichnis*. Speicher-Pools mit alternativem Root-Verzeichnis gelten nur temporär für einen Systemneustart, und alle Einhängepunkte werden relativ zum neuen Root-Verzeichnis des betreffenden Pools geändert.

Erstellen von ZFS-Speicher-Pools mit alternativem Root-Verzeichnis

Speicher-Pools mit alternativem Root-Verzeichnis werden am häufigsten für Wechsel-Datenträger verwendet. In einem solchen Fall wird ein einziges Dateisystem benötigt, und es soll in jede beliebige Stelle auf dem Zielsystem eingehängt werden können. Wenn mithilfe der Option `zpool create -R` ein Speicher-Pool mit alternativem Root-Verzeichnis erstellt wird, wird der Einhängepunkt des Root-Dateisystems automatisch auf `/` gesetzt, was dem Wert des alternativen Root-Verzeichnisses entspricht.

Im folgenden Beispiel wird ein Pool namens `morpheus` mit `/mnt` als alternativem Root-Verzeichnis erstellt:

```
# zpool create -R /mnt morpheus c0t0d0
# zfs list morpheus
NAME                   USED  AVAIL  REFER  MOUNTPOINT
morpheus              32.5K  33.5G     8K  /mnt
```

Bitte beachten Sie das einzige Dateisystem `morpheus`, dessen Einhängepunkt das alternative Root-Verzeichnis `/mnt` ist. Der auf Festplatte gespeicherte Einhängepunkt ist `/`, und der vollständige Pfad zu `/mnt` wird nur im ursprünglichen Kontext der Pool-Erstellung interpretiert. Dieses Dateisystem kann dann auf einem anderen System unter einem beliebigen Speicher-Pool mit alternativem Root-Verzeichnis exportiert und importiert werden, indem die Syntax `-R` *alternativer Root-Wert* verwendet wird.

```
# zpool export morpheus
# zpool import morpheus
cannot mount '/': directory is not empty
# zpool export morpheus
# zpool import -R /mnt morpheus
# zfs list morpheus
NAME                   USED  AVAIL  REFER  MOUNTPOINT
morpheus              32.5K  33.5G     8K  /mnt
```

Importieren von Speicher-Pools mit alternativem Root-Verzeichnis

Pools können auch mithilfe eines alternativen Root-Verzeichnisses importiert werden. Dies ist in Situationen bei der Datenwiederherstellung nützlich, wo Einhängepunkte nicht im Kontext des aktuellen Root-Verzeichnisses interpretiert werden sollen, sondern relativ zu einem temporären Verzeichnis, in dem Reparaturen ausgeführt werden können. Diese Funktion kann auch beim Einhängen von Wechseldatenträgern verwendet werden, wie im vorherigen Abschnitt beschrieben ist.

Im folgenden Beispiel wird ein Pool namens `morpheus` mit `/mnt` als alternativem Root-Verzeichnis importiert. Es wird vorausgesetzt, dass `morpheus` vorher exportiert wurde.

```
# zpool import -R /a pool
# zpool list morpheus
NAME   SIZE   ALLOC  FREE    CAP  HEALTH  ALTROOT
pool  44.8G    78K  44.7G     0%  ONLINE  /a
# zfs list pool
NAME   USED  AVAIL  REFER  MOUNTPOINT
pool  73.5K  44.1G    21K  /a/pool
```

ZFS-Zugriffsrechtsprofile

Wenn Sie ZFS-Verwaltungsaufgaben ohne das Superuser-Benutzerkonto (Root) durchführen wollen, können Sie zum Ausführen von ZFS-Administrationsaufgaben mithilfe der folgenden Profile eine Rolle annehmen:

- ZFS-Speicherplatzverwaltung – Berechtigung zum Erstellen, Löschen und Ändern von Datenspeichergeräten in einem ZFS-Speicher-Pool
- ZFS-Dateisystemverwaltung – Berechtigung zum Erstellen, Löschen und Ändern von ZFS-Dateisystemen

Weitere Informationen zum Erstellen bzw. Annehmen von Rollen finden Sie im *System Administration Guide: Security Services* .

Zusätzlich zur Arbeit mit RBAC-Rollen für die Administration von ZFS-Dateisystemen empfiehlt sich der Einsatz der delegierten ZFS-Administration für verteilte ZFS-Verwaltungsaufgaben. Weitere Informationen finden Sie in Kapitel 9, „Delegierte ZFS-Administration".

KAPITEL 11

Problembehebung und Pool-Wiederherstellung in Oracle Solaris ZFS

Dieses Kapitel enthält Informationen zum Erkennen und Beseitigen von ZFS-Fehlern. Darüber hinaus werden hier auch Maßnahmen zum Vermeiden von Fehlfunktionen beschrieben.

Dieses Kapitel enthält die folgenden Abschnitte:

Erkennen von ZFS-Fehlern

Da ZFS ein Dateisystem mit Datenträgerverwaltungsfunktionen ist, können in diesem System viele verschiedene Fehler auftreten. In diesem Kapitel werden die verschiedenen Fehler kurz umrissen. Dann wird erläutert, wie sie in einem laufenden System erkannt werden können. Dieses Kapitel schließt mit einer Diskussion zum Beheben von Problemen ab. In ZFS treten drei Haupttypen von Fehlern auf:

- „Fehlende Datenspeichergeräte in einem ZFS-Speicher-Pool“ auf Seite 300
- „Beschädigte Datenspeichergeräte in einem ZFS-Speicher-Pool“ auf Seite 300
- „Beschädigte ZFS-Daten“ auf Seite 301

Bitte beachten Sie, dass in einem einzigen Pool alle drei Fehlertypen auftreten können. Deswegen umfasst eine vollständige Problembehebungsroutine das Finden und Beheben eines Fehlers, Weitergehen zum nächsten Fehler usw.

Fehlende Datenspeichergeräte in einem ZFS-Speicher-Pool

Wenn ein Datenspeichergerät vollständig aus dem System entfernt wird, erkennt ZFS, dass es nicht geöffnet werden kann und versetzt es in den Zustand REMOVED. Je nach der Datenreplikationsebene des betreffenden Pools kann es sein, dass durch das Entfernen der gesamte Pool nicht mehr zur Verfügung steht. Bei der Entfernung eines Datenträgers in Konfigurationen mit Datenspiegelung oder RAID-Z bleibt der Pool weiterhin verfügbar. Ein Pool kann in den Zustand `FAULTED` versetzt werden, was bedeutet, dass Daten erst dann wieder verfügbar sind, wenn das Gerät wieder eingebunden wurde. Dies gilt unter folgenden Bedingungen:

- wenn alle Komponenten einer Spiegelung entfernt werden
- wenn mehrere Geräte in einem RAID-Z-Gerät (`raidz1`) entfernt werden
- wenn ein Gerät der obersten Hierarchieebene in einer Konfiguration mit einer Festplatte entfernt wird

Beschädigte Datenspeichergeräte in einem ZFS-Speicher-Pool

Der Begriff ?beschädigt“ deckt eine breite Vielfalt möglicher Fehler ab. Dazu werden beispielsweise die folgenden Fehler gezählt:

- vorübergehende E/A-Fehler aufgrund eines fehlerhaften Datenträgers bzw. Controllers
- Datenbeschädigung auf Datenträgern aufgrund kosmischer Strahlung
- Treiberfehler, wegen denen Daten falsch transportiert werden
- versehentliches Überschreiben von Bereichen auf einem physischen Datenträger durch einen Benutzer

In einigen Fällen treten solche Fehler nur vorübergehend auf, so z. B. bei sporadischen E/A-Fehlern aufgrund eines Controller-Problems. In anderen Fällen ist der Schaden bleibend, wie z. B. bei der Datenbeschädigung auf einem Datenträger. Auch wenn der Schaden bleibend ist, heißt das nicht, dass der Fehler wiederholt auftritt. Wenn ein Administrator beispielsweise versehentlich Bereiche eines Datenträgers überschreibt, ist kein Hardwareausfall aufgetreten, und das Datenspeichergerät muss nicht ausgetauscht werden. Genau festzustellen, mit welchem Problem das Datenspeichergerät behaftet ist, stellt keine leichte Aufgabe dar und wird später eingehender behandelt.

Beschädigte ZFS-Daten

Datenbeschädigung tritt auf, wenn sich Gerätefehler (die auf eines oder mehrere fehlende bzw. beschädigte Datenspeichergeräte hinweisen) auf virtuelle Geräte der obersten Hierarchieebene auswirken. So können beispielsweise in einer Hälfte einer Datenspiegelungskonfiguration Tausende Gerätefehler auftreten, ohne dass dies zur Datenbeschädigung führt. Wenn in der anderen Hälfte der Datenspiegelungskonfiguration an genau der gleichen Stelle ein Fehler auftritt, verursacht das eine Datenbeschädigung.

Eine Datenbeschädigung ist stets bleibend und muss bei der Reparatur besonders berücksichtigt werden. Auch wenn die betreffenden Datenspeichergeräte repariert oder ausgetauscht werden, sind die ursprünglichen Daten für immer verloren. Meist müssen Daten in solchen Situationen aus Sicherungskopien wiederhergestellt werden. Datenfehler werden beim Auftreten protokolliert und können durch regelmäßige Pool-Bereinigung (siehe folgender Abschnitt) in Grenzen gehalten werden. Beim Entfernen eines beschädigten Datenblocks erkennt der nächste Durchlauf der Datenträgerbereinigung, dass die Datenbeschädigung nicht mehr auf dem System vorhanden ist und entfernt die Protokollierung des Fehlers vom System.

Überprüfen der Integrität des ZFS-Dateisystems

Für ZFS gibt es kein Dienstprogramm wie `fsck`. Dieses Dienstprogramm diente üblicherweise zur Reparatur und Validierung von Dateisystemen.

Reparatur von Dateisystemen

Bei herkömmlichen Dateisystemen ist die Art und Weise des Schreibens von Daten von Natur aus anfällig für unerwartete Ausfälle, die zu Inkonsistenzen im Dateisystem führen. Da herkömmliche Dateisysteme nicht transaktionsorientiert sind, können unreferenzierte Datenblöcke, ungültige Verknüpfungszähler oder andere inkonsistente Dateisystemstrukturen auftreten. Mit der Einführung des so genannten Journaling wurden zwar einige dieser Probleme behoben, es können jedoch neue Probleme auftreten, wenn Transaktionen nicht rückgängig gemacht werden können. Inkonsistente Daten in einer ZFS-Konfiguration können nur bei Hardware-Ausfällen (was durch redundante Pools vermieden werden kann) oder bei Fehlern in der ZFS-Software auftreten.

Das Dienstprogramm `fsck` beseitigt bekannte Probleme, von denen UFS-Dateisysteme betroffen sind. Die meisten Probleme, von denen ZFS-Speicher-Pools betroffen sind, sind auf ausgefallene Hardware oder Stromausfälle zurückzuführen. Viele Probleme können durch redundante Pools vermieden werden. Wenn Ihr Pool durch ausgefallene Hardware oder einen Stromausfall beschädigt wurde, gehen Sie wie unter „Reparieren von Schäden am gesamten ZFS-Speicher-Pool“ auf Seite 324 beschrieben vor.

Wenn ein Pool nicht redundant ist, besteht immer das Risiko, dass Sie nach einer Beschädigung des Dateisystems nicht mehr auf Ihre Daten zugreifen können.

Validierung von Dateisystemen

Neben der Dateisystemreparatur stellt das Dienstprogramm `fsck` sicher, dass Daten auf einem Datenträger fehlerfrei sind. Diese Validierung wird üblicherweise durch Aushängen des betreffenden Dateisystems und Ausführen des Dienstprogramms `fsck` durchgeführt. Während dieses Vorgangs muss das System möglicherweise in den Einzelbenutzermodus gebracht werden. Daraus resultieren Ausfallzeiten, die proportional zur Größe des zu überprüfenden Dateisystems sind. Statt eines Dienstprogramms zum expliziten Ausführen der entsprechenden Überprüfungen besitzt ZFS einen Mechanismus zur regelmäßigen Überprüfung auf Inkonsistenzen. Diese als *Bereinigung* bezeichnete Funktion wird häufig in Arbeitsspeichern und anderen Systemen als Methode des Erkennens und Vermeidens von Problemen eingesetzt, die Hardwareausfälle oder Softwarefehlfunktionen verursachen.

Kontrollieren der ZFS-Datenbereinigung

Wenn ZFS einen Fehler erkennt (der auf die Bereinigung oder den Zugriff auf Dateien zurückzuführen ist), wird dieser intern protokolliert, sodass Sie einen schnellen Überblick über alle bekannten Fehler im Pool erhalten.

Explizite ZFS-Datenbereinigung

Die einfachste Methode zum Überprüfen der Datenintegrität besteht im Durchführen einer expliziten Bereinigung aller im Pool enthaltenen Daten. Diesem Vorgang werden alle Daten im Pool einmalig unterzogen, wodurch sichergestellt wird, dass alle Datenblöcke gelesen werden können. Die Bereinigung erfolgt so schnell, wie es das entsprechende Datenspeichergerät zulässt, obwohl E/A-Vorgänge eine niedrigere Priorität als normale Prozessen haben. Dieser Vorgang kann sich negativ auf die Systemleistung auswirken, obgleich die Daten des Pools während der Bereinigung weiterhin verwendbar und weitgehend verfügbar bleiben sollten. Explizite Bereinigungen können mit dem Befehl `zpool scrub` ausgeführt werden. Beispiel:

```
# zpool scrub tank
```

Der Status der aktuellen Bereinigung kann mithilfe des Befehls `zpool status` angezeigt werden. Beispiel:

```
# zpool status -v tank
  pool: tank
 state: ONLINE
 scrub: scrub completed after 0h7m with 0 errors on Tue Tue Feb  2 12:54:00 2010
config:
        NAME        STATE     READ WRITE CKSUM
```

```
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
```

Pro Pool kann immer nur eine aktive Bereinigung ausgeführt werden.

Mithilfe der Option -s können Sie eine laufende Bereinigung stoppen. Beispiel:

```
# zpool scrub -s tank
```

In den meisten Fällen sollte eine Bereinigung vollständig abgeschlossen werden, um die Datenintegrität sicherzustellen. Falls sich eine Bereinigung negativ auf die Systemleistung auswirkt, können Sie sie jederzeit abbrechen.

Durch regelmäßige Bereinigungen wird kontinuierlicher E/A-Datenverkehr zu allen Datenträgern des Systems gewährleistet. Ein Nebeneffekt der Bereinigung besteht darin, dass die Stromverwaltung im Leerlauf befindliche Datenträger nicht in den Energiesparmodus schalten kann. Wenn das System E/A-Vorgänge jederzeit und ohne größere Störungen ausführen kann oder Stromverbrauch kein Problem ist, kann dieser Effekt problemlos ignoriert werden.

Weitere Informationen zur Interpretation der Ausgabe des Befehls `zpool status` finden Sie unter „Abfragen des Status von ZFS-Speicher-Pools“ auf Seite 106.

ZFS-Datenbereinigung und Resilvering

Nach dem Ersetzen eines Datenspeichergeräts wird ein so genanntes ?Resilvering“ (Wiederaufspielen von Daten) durchgeführt, um Daten von unbeschädigten Kopien auf den neuen Datenträger zu kopieren. Dieser Vorgang ist eine Form der Datenträgerbereinigung. Aus diesem Grunde kann in einem Pool immer nur ein solcher Vorgang stattfinden. Ist eine Bereinigung im Gange, wird sie durch das Resilvering unterbrochen und nach Abschluss des Resilvering fortgesetzt.

Weitere Informationen zum Resilvering finden Sie unter „Anzeigen des Resilvering-Status“ auf Seite 320.

Beheben von Problemen mit ZFS

In den folgenden Abschnitten wird beschrieben, wie Sie Probleme mit ZFS-Dateisystem oder Speicher-Pools erkennen und beheben können:

- „Ermitteln, ob in einem ZFS-Speicher-Pool Probleme vorhanden sind“ auf Seite 305
- „Überprüfen der Ausgabe des Befehls `zpool status`“ auf Seite 305
- „Systemprotokoll mit ZFS-Fehlermeldungen“ auf Seite 308

Die folgenden Leistungsmerkmale dienen zur Problemerkennung in ZFS-Konfigurationen:

- Mithilfe des Befehls `zpool status` können ausführliche Informationen zum ZFS-Speicher-Pool angezeigt werden.
- Pool- und Gerätefehler werden mit ZFS/FMA-Diagnosemeldungen gemeldet.
- Frühere ZFS-Befehle, durch die Informationen zum Pool-Status geändert wurden, können mithilfe des Befehls `zpool history` angezeigt werden.

Die meisten ZFS-Probleme können mithilfe des Befehls `zpool status` erkannt werden. Mithilfe dieses Befehls werden verschiedene Fehlfunktionen im System analysiert, die wichtigsten Probleme erkannt und Empfehlungen zu Abhilfemaßnahmen sowie Verweise auf entsprechende Artikel in der Sun Knowledge Base angezeigt. Beachten Sie, dass der Befehl nur ein einziges Problem im Pool erkennen kann, obwohl mehrere Probleme vorhanden sein können. Bei Datenbeschädigungsfehlern wird beispielsweise stets vorausgesetzt, dass ein Datenspeichergerät ausgefallen ist. Durch den Austausch des ausgefallenen Geräts werden jedoch möglicherweise nicht alle Datenbeschädungsprobleme behoben.

Außerdem diagnostiziert und meldet ein ZFS-Diagnoseprogramm Pool- und Datenträgerausfälle. Darüber hinaus werden mit solchen Ausfällen im Zusammenhang stehende Prüfsummen-, E/A-, Geräte- und Poolfehler gemeldet. Von `fmd` gemeldete ZFS-Fehler werden auf der Konsole angezeigt und in der Systemprotokolldatei festgehalten. In den meisten Fällen verweist Sie die `fmd`-Meldung auf den Befehl `zpool status`, mit dessen Hilfe Sie das Problem weiter verfolgen können.

Der grundlegende Problembehebungsvorgang läuft wie folgt ab:

- Suchen Sie, falls möglich, mit dem Befehl `zpool history` die früheren ZFS-Befehle, die vor Auftreten des Problems ausgeführt wurden. Beispiel:

  ```
  # zpool history tank
  History for 'tank':
  2010-07-15.12:06:50 zpool create tank mirror c0t1d0 c0t2d0 c0t3d0
  2010-07-15.12:06:58 zfs create tank/erick
  2010-07-15.12:07:01 zfs set checksum=off tank/erick
  ```

  Beachten Sie, dass in dieser Befehlsausgabe die Prüfsummen für das Dateisystem `tank/erick` deaktiviert sind. Diese Konfiguration wird nicht empfohlen.
- Suchen Sie die Fehler in den `fmd`-Meldungen, die an der Systemkonsole bzw. in der Datei unter `/var/adm/messages` angezeigt werden.
- Weitere Reparaturanweisungen finden Sie mithilfe des Befehls `zpool status -x`.
- Beheben Sie die Probleme, indem Sie wie folgt vorgehen:
  - Ersetzen Sie das ausgefallen oder fehlende Gerät durch ein neues Gerät, und setzen Sie das neue Gerät in Betrieb.
  - Stellen Sie mithilfe einer Sicherungskopie die fehlerhafte Konfiguration bzw. die beschädigten Daten wieder her.
  - Überprüfen Sie die Wiederherstellung mithilfe des Befehls `zpool status - x`.

- Erstellen Sie eine Sicherungskopie der wiederhergestellten Konfiguration (falls möglich).

In diesem Abschnitt wird beschrieben, wie Sie die Ausgabe des Befehls `zpool status` interpretieren, damit Sie Fehler diagnostizieren können. Obwohl die meisten Aufgaben automatisch mithilfe des Befehls ausgeführt werden, müssen Sie genau wissen, um welche Probleme es sich handelt, damit Sie den Ausfall diagnostizieren können. In den nachfolgenden Abschnitten wird beschrieben, wie Sie verschiedenen vorgefundene Probleme beheben können.

Ermitteln, ob in einem ZFS-Speicher-Pool Probleme vorhanden sind

Mit dem Befehl `zpool status -x` können Sie am einfachsten herausfinden, ob in einem System Probleme vorliegen. Mithilfe dieses Befehls werden nur Pools angezeigt, die problembehaftet sind. Wenn in einem System alle Pools ordnungsgemäß funktionieren, wird nur Folgendes angezeigt:

```
# zpool status -x
all pools are healthy
```

Ohne das Flag `-x` werden mithilfe des Befehls die gesamten Statusinformationen aller Pools (oder eines in der Befehlszeile angegebenen Pools) angezeigt, auch wenn diese ordnungsgemäß funktionieren.

Weitere Informationen zu Befehlszeilenoptionen des Befehls `zpool status` finden Sie unter „Abfragen des Status von ZFS-Speicher-Pools“ auf Seite 106.

Überprüfen der Ausgabe des Befehls `zpool status`

Die gesamte Ausgabe des Befehls `zpool status` sieht ungefähr wie folgt aus:

```
# zpool status tank
# zpool status tank
  pool: tank
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t0d0  ONLINE       0     0     0
```

```
            c1t1d0  UNAVAIL      0     0     0  cannot open

errors: No known data errors
```

Es folgt eine Beschreibung dieser Ausgabe:

Gesamtinformationen zum Pool-Status

Dieser Abschnitt der Ausgabe des Befehls `zpool status` enthält die folgenden Felder (einige dieser Felder werden nur angezeigt, wenn im Pool Probleme auftreten):

`pool` Gibt den Namen des Pools an.

`state` Zeigt den aktuellen Funktionsstatus des Pools an. Diese Informationen beziehen sich lediglich auf die Fähigkeit des Pools, für die erforderliche Replikation zu sorgen.

`status` Beschreibt, was mit dem Pool nicht in Ordnung ist. Dieses Feld wird nicht angezeigt, wenn keine Fehler gefunden wurden.

`action` Eine empfohlene Aktion zur Fehlerbehebung. Dieses Feld wird nicht angezeigt, wenn keine Fehler gefunden wurden.

`see` Verweist auf einen Artikel in der Sun Knowledge Base, der ausführliche Reparaturinformationen enthält. Online-Artikel werden öfter als dieses Handbuch aktualisiert und enthalten stets die aktuellsten Reparaturanweisungen. Dieses Feld wird nicht angezeigt, wenn keine Fehler gefunden wurden.

`scrub` Zeigt den aktuellen Status einer Bereinigung an (Datum und Uhrzeit der letzten Bereinigung, Informationen zu einer laufenden Bereinigung, Informationen zur Anforderung von Bereinigungen).

`errors` Zeigt bekannte bzw. unbekannte Datenfehler an.

Pool-Konfigurationsinformationen

Das Feld `config` in der Ausgabe des Befehls `zpool status` beschreibt die Konfigurationsstruktur der Datenspeichergeräte, die den Pool bilden, sowie deren Status und die von diesen Geräten herrührenden Fehler. Der Status kann die folgenden Werte annehmen: `ONLINE`, `FAULTED`, `DEGRADED`, `UNAVAIL` oder `OFFLINE`. Wenn der Status eines Pools nicht `ONLINE` ist, wurde die Fehlertoleranz des Pools eingeschränkt.

Im zweiten Abschnitt der Konfigurationsinformationen wird die Fehlerstatistik angezeigt. Diese Fehler werden in drei Kategorien eingeteilt:

- `READ` – E/A-Fehler während der Ausgabe einer Leseanforderung
- `WRITE` – E/A-Fehler während der Ausgabe einer Schreibanforderung
- `CKSUM` – Prüfsummenfehler, d. h., das Gerät hat nach einer Leseanforderung beschädigte Daten zurückgegeben.

Mithilfe dieser Kategorien kann ermittelt werden, ob es sich um bleibende Schäden handelt. Eine geringe Anzahl auftretender E/A-Fehler kann die Folge zeitweiliger Ausfälle sein, während eine größere Anzahl an E/A-Fehlern auf ein bleibendes Problem mit dem entsprechenden Gerät hinweisen kann. Bei diesen Fehlern muss es sich nicht unbedingt um Datenbeschädigung handeln, auch wenn sie von den Anwendungen so interpretiert werden. Wenn das Gerät zu einer redundanten Konfiguration gehört, kann es sein, dass die Datenträger nicht behebbare Fehler aufweisen, während auf der RAID-Z- bzw. Datenspiegelungsebene keine Fehler angezeigt werden. In solchen Fällen hat ZFS die unbeschädigten Daten erfolgreich abgerufen und versucht, die beschädigten Daten durch vorhandene Replikationen zu ersetzen.

Weitere Informationen zur Interpretation dieser Fehler finden Sie unter „Ermitteln des Gerätefehlertyps“ auf Seite 311.

Zusätzliche hilfreiche Informationen werden in der letzten Spalte der Ausgabe des Befehls `zpool status` angezeigt. Diese Information ergänzen die im Feld `state` enthaltenen Informationen und helfen bei der Diagnose von Fehlern. Bei Datenspeichergeräten mit dem Status `FAULTED` zeigt dieses Feld an, ob auf das betreffende Gerät zugegriffen werden kann oder die Daten auf dem Gerät beschädigt sind. Wenn auf das Datenspeichergerät mithilfe von Resilvering Daten neu aufgespielt werden, zeigt dieses Feld den Verlauf dieses Vorgangs an.

Weitere Informationen zur Überwachung des Resilvering-Vorgangs finden Sie in „Anzeigen des Resilvering-Status“ auf Seite 320.

Status eines Bereinigungsvorgangs

Im Bereinigungsabschnitt der Ausgabe des Befehls `zpool status` wird der aktuelle Status von Bereinigungsvorgängen angezeigt, die explizit ausgeführt werden. Diese Informationen weisen nicht auf Fehler hin, die im System aufgetreten sind, können aber zum Ermitteln der Genauigkeit des Meldens von Datenbeschädigungsfehlern herangezogen werden. Wenn die letzte Bereinigung erst vor kurzem ausgeführt wurde, sind Datenbeschädigungen höchstwahrscheinlich bereits bekannt.

Meldungen zum Abschluss von Bereinigungen bleiben nach Systemneustarts erhalten.

Weitere Informationen zur Datenbereinigung und zur Interpretation dieser Informationen finden Sie unter „Überprüfen der Integrität des ZFS-Dateisystems“ auf Seite 301.

Datenbeschädigungsfehler

Der Befehl `zpool status` zeigt auch an, ob im Zusammenhang mit einem Pool bekannte Fehler aufgetreten sind. Diese Fehler können während der Datenbereinigung oder im Normalbetrieb gefunden worden sein. ZFS führt ein kontinuierliches Protokoll aller mit einem Pool im Zusammenhang stehenden Datenfehler. Nach jedem Abschluss einer vollständigen Systembereinigung wird das Protokoll entsprechend aktualisiert.

Datenbeschädigungsfehler sind stets schwerwiegend. Ihr Vorhandensein weist darauf hin, dass bei mindestens einem Anwendungsprogramm aufgrund beschädigter Daten im Pool ein E/A-Fehler aufgetreten ist. Gerätefehler innerhalb eines redundanten Pools verursachen keine Datenbeschädigung und werden in diesem Protokoll nicht festgehalten. Standardmäßig wird nur die Anzahl der gefundenen Fehler angezeigt. Eine vollständige Liste mit Fehlern und deren Informationen kann mit dem Befehl `zpool status -v` angezeigt werden. Beispiel:

```
# zpool status -v
  pool: tank
 state: UNAVAIL
status: One or more devices are faulted in response to IO failures.
action: Make sure the affected devices are connected, then run 'zpool clear'.
   see: http://www.sun.com/msg/ZFS-8000-HC
 scrub: scrub completed after 0h0m with 0 errors on Tue Feb  2 13:08:42 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        UNAVAIL      0     0     0  insufficient replicas
          c1t0d0    ONLINE       0     0     0
          c1t1d0    UNAVAIL      4     1     0  cannot open

errors: Permanent errors have been detected in the following files:

/tank/data/aaa
/tank/data/bbb
/tank/data/ccc
```

Eine ähnliche Meldung wird auch von `fmd` auf der Systemkonsole angezeigt und in der Datei `/var/adm/messages` protokolliert. Diese Meldungen können auch mit dem Befehl `fmdump` verfolgt werden.

Weitere Informationen zur Interpretation von Datenbeschädigungsfehlern finden Sie unter „Ermitteln der Art der Datenbeschädigung“ auf Seite 322.

Systemprotokoll mit ZFS-Fehlermeldungen

Neben der kontinuierlichen Verfolgung von Fehlern innerhalb eines Pools zeigt ZFS beim Auftreten bestimmter Ereignisse auch Systemprotokollmeldungen an. In den folgenden Situationen werden Ereignisse zur Benachrichtigung des Administrators ausgelöst:

- **Statusübergang eines Datenspeichergeräts** – Wenn ein Datenspeichergerät in den Status `FAULTED` übergeht, protokolliert ZFS eine Meldung, die darauf hinweist, dass die Fehlertoleranz des Pools beeinträchtigt werden könnte. Eine ähnliche Meldung wird gesendet, wenn das Gerät später wieder in Betrieb genommen und die ordnungsgemäße Pool-Funktion wiederhergestellt wird.

- **Datenbeschädigung** – Beim Erkennen von Datenbeschädigungen protokolliert ZFS eine Meldung, die beschreibt, wann und wo die Datenbeschädigung erkannt wurde. Diese Meldung wird nur beim allerersten Auftreten des Ereignisses protokolliert. Ein nachfolgendes Auftreten dieser Ereignisses löst keine Meldung mehr aus.
- **Pool- und Geräteausfälle** – Wenn ein Pool oder Gerät ausfällt, meldet der Fehlerverwaltungsdämon diese Fehler mithilfe von Systemprotokollmeldungen und mit dem Befehl `fmdump`.

Wenn ZFS einen Gerätefehler erkennt und diesen automatisch behebt, wird keine Benachrichtigung gesendet. Solche Fehler stellen keine Einschränkung der Pool-Redundanz bzw. Datenintegrität dar und sind darüber hinaus normalerweise die Folge eines Treiberproblems mit eigenen entsprechenden Fehlermeldungen.

Reparieren einer beschädigten ZFS-Konfiguration

ZFS unterhält im Root-Dateisystem einen Cache aktiver Pools und ihrer Konfiguration. Wenn diese Cache-Datei beschädigt wird bzw. nicht mehr mit den auf dem Datenträger gespeicherten Konfigurationsinformationen übereinstimmt, kann auf einen Pool nicht mehr zugegriffen werden. ZFS versucht, eine solche Situation zu vermeiden, obwohl aufgrund der Eigenschaften des zugrunde liegenden Speichers immer die Möglichkeit besteht, dass Daten beschädigt werden können. Solche Situationen führen normalerweise zu einem Verschwinden eines ansonsten verfügbaren Pools aus dem System und können sich auch in unvollständigen Konfigurationen manifestieren, denen eine unbekannte Anzahl an virtuellen Geräten der obersten Hierarchie fehlt. In allen Fällen kann die Konfiguration durch Exportieren des Pools (falls er zugänglich ist) und anschließendes Importieren wiederhergestellt werden.

Weitere Informationen zum Importieren und Exportieren von Pools finden Sie unter „Migrieren von ZFS-Speicher-Pools“ auf Seite 115.

Abhilfe bei Nichtverfügbarkeit eines Geräts

Wenn auf ein Gerät nicht zugegriffen werden kann, wird es in der Ausgabe des Befehls `zpool status` mit dem Status `UNAVAIL` angezeigt. Dieser Status bedeutet, dass ZFS beim ersten Zugriff auf den Pool nicht auf das betreffende Datenspeichergerät zugreifen konnte oder es seitdem nicht mehr verfügbar ist. Wenn durch dieses Datenspeichergerät ein virtuelles Gerät der obersten Hierarchieebene nicht mehr verfügbar ist, können von diesem Pool keine Daten abgerufen werden. Andernfalls kann die Fehlertoleranz des Pools beeinträchtigt werden. In jedem Fall muss das Gerät zum Wiederherstellen des Normalbetriebs wieder in das System integriert werden.

Nach einem Geräteausfall wird von `fmd` in etwa die folgende Meldung angezeigt:

```
SUNW-MSG-ID: ZFS-8000-FD, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Thu Jun 24 10:42:36 PDT 2010
PLATFORM: SUNW,Sun-Fire-T200, CSN: -, HOSTNAME: neo2
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: a1fb66d0-cc51-cd14-a835-961c15696fcb
DESC: The number of I/O errors associated with a ZFS device exceeded
acceptable levels.  Refer to http://sun.com/msg/ZFS-8000-FD for more information.
AUTO-RESPONSE: The device has been offlined and marked as faulted.  An attempt
will be made to activate a hot spare if available.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Run 'zpool status -x' and replace the bad device.
```

Um ausführlichere Informationen zu Geräteproblemen und deren Behebung anzuzeigen, verwenden Sie den Befehl `zpool status -x`. Beispiel:

```
# zpool status -x
  pool: tank
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: scrub completed after 0h0m with 0 errors on Tue Feb  2 13:15:20 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  UNAVAIL      0     0     0  cannot open

errors: No known data errors
```

Diese Befehlsausgabe zeigt, dass das fehlende Gerät `c1t1d0` nicht funktioniert. Wenn das Gerät fehlerhaft ist, ersetzen Sie es.

Nehmen Sie das ersetzte Gerät dann mit dem Befehl `zpool online` in Betrieb. Beispiel:

```
# zpool online tank c1t1d0
```

Als Letztes stellen Sie sicher, dass der Pool mit dem ersetzten Datenspeichergerät ordnungsgemäß funktioniert. Beispiel:

```
# zpool status -x tank
pool 'tank' is healthy
```

Wiedereinbinden eines Datenspeichergeräts

Die Art und Weise, wie ein fehlendes Datenspeichergerät wieder in ein System eingebunden wird, hängt vom jeweiligen Gerät ab. Wenn auf das Gerät über das Netzwerk zugegriffen wird, muss die Netzwerkverbindung wiederhergestellt werden. Wenn das Datenspeichergerät ein

USB-Gerät oder ein anderer Wechseldatenträger ist, muss es wieder an das System angeschlossen werden. Wenn es sich bei dem betreffenden Gerät um eine lokale Festplatte handelt, kann es sein, dass das Gerät aufgrund eines Controller-Ausfalls nicht mehr vom System erkannt wird. In diesem Fall muss der Controller ausgetauscht werden, wonach die betreffenden Datenträger wieder verfügbar sind. Es können noch weitere Probleme vorliegen, die mit der der Art der Hardware und ihrer Konfiguration in Zusammenhang stehen. Wenn ein Laufwerk ausfällt und vom System nicht mehr erkannt wird, muss das Datenspeichergerät als beschädigt eingestuft werden. Folgen Sie der unter „Ersetzen oder Reparieren eines beschädigten Geräts“ auf Seite 311 beschriebenen Vorgehensweise.

Benachrichtigung von ZFS nach Wiederherstellung der Verfügbarkeit

Nach dem Wiedereinbinden eines Datenspeichergeräts in das System erkennt ZFS unter Umständen seine Verfügbarkeit automatisch. Dies muss aber nicht so ein. Wenn der Pool ausgefallen war oder das System während der Wiedereinbindung neu gestartet wurde, sucht ZFS automatisch alle Datenspeichergeräte ab, während es versucht, auf den Pool zuzugreifen. Wenn der Pool in seiner Funktionstüchtigkeit beeinträchtigt war und das Gerät während des Systembetriebs ausgetauscht wurde, müssen Sie ZFS mithilfe des Befehls `zpool online` benachrichtigen, dass das Gerät jetzt wieder verfügbar ist und wieder auf das Gerät zugegriffen werden kann. Beispiel:

```
# zpool online tank c0t1d0
```

Weitere Informationen zum Inbetriebnehmen von Geräten finden Sie unter „Inbetriebnehmen eines Gerätes“ auf Seite 94.

Ersetzen oder Reparieren eines beschädigten Geräts

In diesem Abschnitt wird beschrieben, wie die verschiedenen Fehlertypen eines Datenspeichergerätes ermittelt, vorübergehende Fehler gelöscht und Geräte ausgetauscht werden können.

Ermitteln des Gerätefehlertyps

Der Begriff *beschädigtes Gerät* ist nicht klar umrissen und kann verschiedene mögliche Situationen beschreiben:

- **Bitfäule** – Mit der Zeit können äußere Einflüsse wie Magnetfelder und kosmische Strahlung dazu führen, dass auf Datenträgern gespeicherte Bits unvorhergesehene Werte annehmen. Solche Ereignisse sind relativ selten, treten aber häufig genug auf, um potenzielle Datenbeschädigung in größeren und lange laufenden Systemen zu verursachen.
- **Fehlgeleitete Lese- oder Schreibvorgänge** – Firmware- oder Hardwarefehler können dazu führen, dass Lese- oder Schreibvorgänge ganzer Datenblöcke auf den falschen Bereich auf dem Datenträger verweisen. Diese Fehler sind normalerweise vorübergehend, obwohl eine große Anzahl solcher Fehler auf ein fehlerhaftes Laufwerk hinweisen kann.
- **Administratorfehler** – Administratoren können versehentlich Datenträgerbereiche mit ungültigen Daten überschreiben (z. B. das Kopieren von `/dev/zero` auf bestimmte Datenträgerbereiche) und so Daten auf dem Datenträger dauerhaft beschädigen. Solche Fehler sind stets vorübergehend.
- **Zeitweilige Ausfälle** – Datenträger können zeitweilig ausfallen, wodurch E/A-Vorgänge fehlschlagen. Diese Situation tritt normalerweise bei Datenspeichergeräten auf, auf die über das Netzwerk zugegriffen wird, obwohl auch bei lokalen Datenträgern solche Ausfälle auftreten können. Solche Fehler sind können vorübergehend sein.
- **Fehlerhafte bzw. unzuverlässig arbeitende Hardware** – Unter diese Kategorie fallen alle durch fehlerhafte Hardware verursachten Probleme. Dies können dauerhafte E/A-Fehler, falscher Datentransport und daraus folgende regellose Datenbeschädigung sowie eine Reihe anderer Fehler sein. Solche Fehler sind normalerweise dauerhaft.
- **Außer Betrieb genommene Datenspeichergeräte** – Wenn ein Gerät außer Betrieb genommen wurde, wird angenommen, dass es der Administrator in diesen Zustand versetzt hat, weil es fehlerhaft ist. Der Administrator, der das Gerät in diesen Zustand versetzt hat, kann überprüfen, ob diese Annahme richtig ist.

Die genaue Ermittlung von Fehlerursachen kann sich schwierig gestalten. Der erste Schritt besteht darin, die Fehlerzähler in der Ausgabe des Befehls `zpool status` zu überprüfen. Beispiel:

```
# zpool status -v tpool
  pool: tpool
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: http://www.sun.com/msg/ZFS-8000-8A
 scrub: scrub completed after 0h0m with 2 errors on Tue Jul 13 11:08:37 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tpool       ONLINE       2     0     0
          c1t1d0    ONLINE       2     0     0
          c1t3d0    ONLINE       0     0     0
errors: Permanent errors have been detected in the following files:
```

```
        /tpool/words
```

Die Fehler werden in E/A- und Prüfsummenfehler unterteilt. Daraus lässt sich unter Umständen auf den Fehlertyp schließen. Im Normalbetrieb treten während einer langen Systemlaufzeit nur einige wenige Fehler auf. Wenn eine hohe Anzahl von Fehlern angezeigt wird, deutet dies wahrscheinlich auf einen bevorstehenden bzw. vollständigen Geräteausfall hin. Ein Administratorfehler kann jedoch ebenfalls zu hohen Fehleranzahlen führen. Eine weitere Informationsquelle ist das Systemprotokoll. Wenn das Protokoll eine große Anzahl an Meldungen von SCSI- bzw. Fibre Channel-Treibern enthält, weist dies möglicherweise auf schwerwiegende Hardwareprobleme hin. Wenn keine Systemmeldungen protokolliert werden, ist der Schaden wahrscheinlich vorübergehend.

Das Ziel besteht in der Beantwortung der folgenden Frage:

Ist es wahrscheinlich, dass an diesem Gerät wieder ein Fehler auftritt?

Nur einmal auftretende Fehler werden als *vorübergehend* eingestuft und ziehen keine potenziellen Ausfälle nach sich. Fehler, die dauerhaft oder ernstlich genug sind, um potenzielle Hardwareausfälle auszulösen, werden als *schwerwiegend* eingestuft. Die Ermittlung von Fehlertypen sprengt den Rahmen der gegenwärtig mit ZFS verfügbaren automatisierten Softwarelösungen und muss manuell vom Administrator durchgeführt werden. Nach der Ermittlung des Fehlertyps können entsprechende Abhilfemaßnahmen ergriffen werden. Diese bestehen entweder im Löschen vorübergehender Fehler oder im Austauschen des betreffenden Datenspeichergeräts aufgrund schwerwiegender Fehler. Die entsprechenden Reparaturvorgänge werden in den nächsten Abschnitten erläutert.

Auch wenn Gerätefehler als vorübergehend eingestuft wurden, können sie trotzdem nicht mehr rückgängig zu machende Datenfehler im Pool verursacht haben. Diese Fehler erfordern auch dann spezielle Reparaturmaßnahmen, wenn das zugrunde liegende Datenspeichergerät ordnungsgemäß funktioniert oder anderweitig repariert wurde. Weitere Informationen zum Beseitigen von Datenfehlern finden Sie unter „Reparieren beschädigter Daten" auf Seite 321.

Löschen vorübergehender Fehler

Wenn Gerätefehler als vorübergehend eingestuft wurden und die zukünftige ordnungsgemäße Funktion des betreffenden Datenspeichergeräts nicht beeinträchtigen, können sie gelöscht werden. Damit wird angezeigt, dass kein schwerwiegender Fehler aufgetreten ist. Mit dem Befehl `zpool clear` können Sie Fehlerzähler für Datenspeichergeräte mit RAID-Z- bzw. Datenspiegelungskonfigurationen zurücksetzen. Beispiel:

```
# zpool clear tank c1t1d0
```

Mithilfe dieser Syntax werden alle Fehler gelöscht und alle Fehlerzähler zurückgesetzt, die mit dem betreffenden Datenspeichergerät in Zusammenhang stehen.

Zum Löschen aller mit den virtuellen Geräten im Pool in Verbindung stehenden Fehler und Zurücksetzen aller Fehlerzähler dient die folgende Syntax:

```
# zpool clear tank
```

Weitere Informationen zum Löschen von Pool-Fehlern finden Sie unter „Löschen von Gerätefehlern im Speicher-Pool“ auf Seite 95.

Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool

Wenn ein Datenspeichergerät dauerhaft beschädigt ist bzw. ein solcher Schaden bevorsteht, muss es ausgetauscht werden. Ob das betreffende Gerät ersetzt werden kann, hängt von der Konfiguration ab.

Ermitteln, ob ein Gerät ausgetauscht werden kann

Damit ein Gerät ausgetauscht werden kann, muss sich der Pool im Status `ONLINE` befinden. Das betreffende Gerät muss zu einer redundanten Konfiguration gehören, oder es muss ordnungsgemäß funktionieren (d. h. sich im Status `ONLINE` befinden). Wenn das Gerät zu einer redundanten Konfiguration gehört, müssen ausreichende Replikationen vorhanden sein, aus denen unbeschädigte Daten wiederhergestellt werden können. Wenn zwei Datenträger in einer vierfachen Datenspiegelungskonfiguration fehlerhaft sind, können beide Datenträger ausgetauscht werden, da gültige Datenreplikationen vorhanden sind. Wenn jedoch zwei Datenträger in einer vierfachen RAID-Z-Konfiguration `raidz1` fehlerhaft sind, kann keiner der beiden Datenträger ausgetauscht werden, da nicht genügend gültige Datenreplikationen verfügbar sind, aus denen Daten wiederhergestellt werden können. Wenn das Gerät beschädigt, aber noch in Betrieb ist, kann es ausgetauscht werden, solange sich der Pool nicht im Status `FAULTED` befindet. Wenn nicht genügend Replikationen mit gültigen Daten verfügbar sind, werden jedoch auch die beschädigten Daten auf das neue Datenspeichergerät kopiert.

In der folgenden Konfiguration kann der Datenträger `c1t1d0` ausgetauscht werden, und die gesamten Daten im Pool werden von der ordnungsgemäßen Replikation `c1t0d0` kopiert.

```
    mirror            DEGRADED
    c1t0d0             ONLINE
    c1t1d0             FAULTED
```

Der Datenträger `c1t0d0` kann ebenfalls ausgetauscht werden, obwohl keine Datenselbstheilung erfolgen kann, da keine ordnungsgemäße Datenreplikation vorhanden ist.

In der folgenden Konfiguration kann keiner der fehlerhaften Datenträger ausgetauscht werden. Die Datenträger mit dem Status `ONLINE` können ebenfalls nicht ausgetauscht werden, da der Pool selbst fehlerhaft ist.

```
    raidz              FAULTED
    c1t0d0             ONLINE
    c2t0d0             FAULTED
    c3t0d0             FAULTED
    c4t0d0             ONLINE
```

In der folgenden Konfiguration können alle Datenträger der obersten Hierarchieebene ausgetauscht werden, obwohl auch auf den alten Datenträgern befindliche ungültige Daten auf den neuen Datenträger kopiert werden.

```
c1t0d0         ONLINE
c1t1d0         ONLINE
```

Wenn alle Datenträger fehlerhaft sind, kann nichts ausgetauscht werden, da dann der Pool selbst fehlerhaft ist.

Datenspeichergeräte, die nicht ausgetauscht werden können

Wenn ein Pool durch den Ausfall eines Datenspeichergeräts fehlerhaft wird oder das betreffende Gerät in einer nicht redundanten Konfiguration zu viele Datenfehler enthält, kann es nicht sicher ausgetauscht werden. Ohne ausreichende Redundanz sind keine entsprechenden gültigen Daten vorhanden, die die Fehler auf dem beschädigten Gerät beseitigen könnten. In einem solchen Fall besteht nur die Möglichkeit, den Pool zu löschen, die Konfiguration neu zu erstellen und die Daten aus einer Sicherungskopie wiederherzustellen.

Weitere Informationen zum Wiederherstellen eines gesamten Pools finden Sie unter „Reparieren von Schäden am gesamten ZFS-Speicher-Pool“ auf Seite 324.

Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool

Wenn Sie ermittelt haben, dass ein Datenspeichergerät ausgetauscht werden kann, können Sie es mithilfe des Befehls `zpool replace` ersetzen. Um ein beschädigtes Gerät durch ein anderes Gerät zu ersetzen, verwenden Sie eine Syntax wie die folgende:

```
# zpool replace tank c1t1d0 c2t0d0
```

Mithilfe dieses Befehls werden Daten vom beschädigten Gerät oder von anderen Geräten im Pool, sofern sich dieser in einer redundanten Konfiguration befindet, auf das neue Gerät migriert. Nach Abschluss des Befehls wird das beschädigte Datenspeichergerät von der Konfiguration abgetrennt und kann dann aus dem System entfernt werden. Wenn Sie das

Datenspeichergerät bereits entfernt und an der gleichen Stelle durch ein neues Gerät ersetzt haben, sollten Sie das Einzelgeräteformat des Befehls verwenden. Beispiel:

```
# zpool replace tank c1t1d0
```

Mithilfe dieses Befehls wird das neue Datenspeichergerät formatiert, und anschließend werden die Daten aus der Konfiguration durch Resilvering aufgespielt.

Weitere Informationen zum Befehl `zpool replace` finden Sie unter „Austauschen von Geräten in einem Speicher-Pool" auf Seite 95.

BEISPIEL 11–1 Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool

Das folgende Beispiel zeigt, wie ein Gerät (`c1t3d0`) in einem Speicher-Pool mit Datenspiegelung `tank` auf einem Sun Fire x4500-System von Oracle ersetzt wird. Um die Festplatte `c1t3d0` durch eine neue Festplatte an derselben Position (`c1t3d0`) ersetzen, müssen Sie die Festplatte zunächst dekonfigurieren. Der Vorgang wird im Wesentlichen wie folgt durchgeführt:

- Nehmen Sie die Festplatte (`c1t3d0`), die ersetzt werden soll, außer Betrieb. Eine in Gebrauch befindliche Festplatte kann nicht dekonfiguriert werden.
- Verwenden Sie den Befehl `cfgadm`, um die betroffene Festplatte (`c1t3d0`) zu ermitteln, und entfernen Sie sie aus der Konfiguration. Der Pool wird durch die außer Betrieb genommene Festplatte in der Datenspiegelungskonfiguration in einen eingeschränkten Zustand versetzt, bleibt aber weiterhin verfügbar.
- Ersetzen Sie die Festplatte physisch (`c1t3d0`). Vergewissern Sie sich vor dem Ausbauen des fehlerhaften Laufwerks, dass die blaue Ausbaubereitschaft-LED leuchtet.
- Konfigurieren Sie die Festplatte (`c1t3d0`) erneut.
- Setzen Sie die neue Festplatte (`c1t3d0`) in Betrieb.
- Führen Sie den Befehl `zpool replace` aus, um die Festplatte (`c1t3d0`) zu ersetzen.

Hinweis – Wenn Sie zuvor die Eigenschaft `autoreplace` des Pools auf `on` gesetzt haben, wird jedes neue Gerät, das sich an der gleichen Stelle befindet, an der sich zuvor ein anderes zum Pool gehörendes Geräts befand, automatisch formatiert und ohne den Befehl `zpool replace` ersetzt. Dieses Leistungsmerkmal wird möglicherweise nicht auf jeder Art von Hardware unterstützt.

- Wenn eine ausgefallene Festplatte automatisch durch eine Hot-Spare-Festplatte ersetzt wurde, müssen Sie die Hot-Spare-Festplatte möglicherweise nach dem Ersetzen der ausgefallenen Festplatte abtrennen. Wenn `c2t4d0` beispielsweise noch immer eine aktive Hot-Spare-Festplatte ist, nachdem die ausgefallene Festplatte ersetzt wurde, trennen Sie sie ab.

BEISPIEL 11–1 Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool *(Fortsetzung)*

```
# zpool detach tank c2t4d0
```

Im folgenden Beispiel wird gezeigt, wie eine Festplatte in einem ZFS-Speicher-Pool ersetzt wird.

```
# zpool offline tank c1t3d0
# cfgadm | grep c1t3d0
sata1/3::dsk/c1t3d0            disk         connected    configured   ok
# cfgadm -c unconfigure sata1/3
Unconfigure the device at: /devices/pci@0,0/pci1022,7458@2/pci11ab,11ab@1:3
This operation will suspend activity on the SATA device
Continue (yes/no)? yes
# cfgadm | grep sata1/3
sata1/3                        disk         connected    unconfigured ok
<Physically replace the failed disk c1t3d0>
# cfgadm -c configure sata1/3
# cfgadm | grep sata1/3
sata1/3::dsk/c1t3d0            disk         connected    configured   ok
# zpool online tank c1t3d0
# zpool replace tank c1t3d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Tue Feb  2 13:17:32 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror-2  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
```

Beachten Sie, dass in der obigen `zpool`-Ausgabe sowohl die neue als auch die alte Festplatte unter *replacing* (wird ersetzt) angezeigt werden kann. Beispiel:

```
replacing     DEGRADED     0     0    0
  c1t3d0s0/o  FAULTED      0     0    0
  c1t3d0      ONLINE       0     0    0
```

Dieser Text bedeutet, dass der Austauschvorgang läuft und an der neuen Festplatte das Resilvering durchgeführt wird.

Um eine Festplatte (`c1t3d0`) durch eine andere Festplatte (`c4t3d0`) zu ersetzen, müssen Sie nur den Befehl `zpool replace` ausführen. Beispiel:

BEISPIEL 11–1 Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool *(Fortsetzung)*

```
# zpool replace tank c1t3d0 c4t3d0
# zpool status
  pool: tank
 state: DEGRADED
 scrub: resilver completed after 0h0m with 0 errors on Tue Feb  2 13:35:41 2010
config:

        NAME             STATE     READ WRITE CKSUM
        tank             DEGRADED     0     0     0
          mirror-0       ONLINE       0     0     0
            c0t1d0       ONLINE       0     0     0
            c1t1d0       ONLINE       0     0     0
          mirror-1       ONLINE       0     0     0
            c0t2d0       ONLINE       0     0     0
            c1t2d0       ONLINE       0     0     0
          mirror-2       DEGRADED     0     0     0
            c0t3d0       ONLINE       0     0     0
            replacing    DEGRADED     0     0     0
              c1t3d0     OFFLINE      0     0     0
              c4t3d0     ONLINE       0     0     0

errors: No known data errors
```

Mitunter muss der Befehl `zpool status` mehrmals ausgeführt werden, bevor der Austauschvorgang abgeschlossen ist.

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Tue Feb  2 13:35:41 2010
config:

        NAME          STATE     READ WRITE CKSUM
        tank          ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c0t1d0    ONLINE       0     0     0
            c1t1d0    ONLINE       0     0     0
          mirror-1    ONLINE       0     0     0
            c0t2d0    ONLINE       0     0     0
            c1t2d0    ONLINE       0     0     0
          mirror-2    ONLINE       0     0     0
            c0t3d0    ONLINE       0     0     0
            c4t3d0    ONLINE       0     0     0
```

BEISPIEL 11–2 Ersetzen eines fehlgeschlagenen Protokolliergeräts

Das folgende Beispiel zeigt die Wiederherstellung des Normalzustands nach dem Ausfall eines Protokolliergeräts (`c0t5d0`) im Speicher-Pool (`pool`). Der Vorgang wird im Wesentlichen wie folgt durchgeführt:

- Überprüfen Sie die Ausgabe des Befehls `zpool status -x` und die FMA-Diagnosemeldung, beschrieben unter:

BEISPIEL 11–2 Ersetzen eines fehlgeschlagenen Protokolliergeräts *(Fortsetzung)*

http://www.sun.com/msg/ZFS-8000-K4

- Ersetzen Sie das fehlgeschlagene Protokolliergerät physisch.
- Nehmen Sie das neue Protokolliergerät in Betrieb.
- Löschen Sie den Fehlerzustand des Pools.

```
# zpool status -x
  pool: pool
 state: FAULTED
status: One or more of the intent logs could not be read.
        Waiting for adminstrator intervention to fix the faulted pool.
action: Either restore the affected device(s) and run 'zpool online',
        or ignore the intent log records by running 'zpool clear'.
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        FAULTED      0     0     0 bad intent log
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
        logs        FAULTED      0     0     0 bad intent log
          c0t5d0    UNAVAIL      0     0     0 cannot open
<Physically replace the failed log device>
# zpool online pool c0t5d0
# zpool clear pool

# zpool status -x
  pool: pool
 state: FAULTED
status: One or more of the intent logs could not be read.
        Waiting for adminstrator intervention to fix the faulted pool.
action: Either restore the affected device(s) and run 'zpool online',
        or ignore the intent log records by running 'zpool clear'.
 scrub: none requested
config:

        NAME          STATE     READ WRITE CKSUM
        pool          FAULTED      0     0     0 bad intent log
          mirror-0    ONLINE       0     0     0
            c0t1d0    ONLINE       0     0     0
            c0t4d0    ONLINE       0     0     0
        logs          FAULTED      0     0     0 bad intent log
          c0t5d0      UNAVAIL      0     0     0 cannot open
<Physically replace the failed log device>
# zpool online pool c0t5d0
# zpool clear pool
```

Anzeigen des Resilvering-Status

Je nach Kapazität des Datenträgers und der Datenmenge im Pool kann das Austauschen eines Datenspeichergeräts geraume Zeit dauern. Der Vorgang des Übertragens von Daten von einem Datenspeichergerät auf ein anderes Gerät nennt man *Resilvering*. Er kann mithilfe des Befehls `zpool status` überwacht werden.

Herkömmliche Dateisysteme kopieren Daten auf Datenblockebene. Da in ZFS die künstliche Schicht des Volume Managers beseitigt wurde, kann das Resilvering hier viel leistungsfähiger und kontrollierter durchgeführt werden. Die beiden Hauptvorteile dieser Funktion bestehen in Folgendem:

- ZFS kopiert beim Resilvering nur die Mindestmenge erforderlicher Daten. Im Falle eines kurzen Ausfalls (im Gegensatz zu einem vollständigen Austausch von Datenspeichergeräten) können Daten innerhalb weniger Minuten bzw. Sekunden auf den gesamten Datenträger aufgespielt werden. Nach dem Austauschen eines Datenträgers ist der Zeitraum, den das Wiederaufspielen von Daten benötigt, proportional zur Datenmenge auf dem betreffenden Datenträger. Der Austausch eines 500-GB-Datenträgers kann in Sekundenschnelle durchgeführt werden, wenn im Pool nur einige wenige GB mit Daten belegt sind.
- Das Resilvering kann unterbrochen werden und ist sicher. Wenn am System ein Stromausfall auftritt oder es neu gestartet wird, fährt der Resilvering-Vorgang an genau der Stelle fort, wo er unterbrochen wurde, ohne dass dazu manuelle Eingriffe erforderlich sind.

Der Status des Resilvering-Vorgangs kann mit dem Befehl `zpool status` angezeigt werden. Beispiel:

```
# zpool status tank
  pool: tank
 state: DEGRADED
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress for 0h0m, 22.60% done, 0h1m to go
config:
        NAME                  STATE     READ WRITE CKSUM
        tank             DEGRADED     0     0     0
          mirror-0       DEGRADED     0     0     0
            replacing-0  DEGRADED     0     0     0
              c1t0d0     UNAVAIL      0     0     0  cannot open
              c2t0d0     ONLINE       0     0     0  85.0M resilvered
            c1t1d0       ONLINE       0     0     0

errors: No known data errors
```

In diesem Beispiel wird der Datenträger `c1t0d0` durch das Gerät `c2t0d0` ersetzt. Dieses Ereignis wird in der Statusausgabe durch das virtuelle Ersatzgerät in der Konfiguration dargestellt. Dieses Gerät ist kein echtes Gerät, und Sie können mit diesem Gerät auch keinen Pool erstellen.

Der Zweck dieses Geräts besteht lediglich darin, den Resilvering-Vorgang anzuzeigen, damit festgestellt werden kann, welcher Datenträger ausgetauscht wird.

Beachten Sie, dass ein Pool, in dem ein Resilvering stattfindet, in den Status ONLINE oder DEGRADED versetzt wird, da er während des Resilvering-Vorgangs nicht das erforderliche Niveau an Datenredundanz gewährleisten kann. Das Resilvering findet so schnell wie möglich statt, obwohl die damit verbundenen E/A-Prozesse stets eine niedrigere Priorität als E/A-Benutzeranforderungen haben, um die Auswirkung auf die Systemleistung zu minimieren. Nach dem Abschluss des Resilvering wird die neue, vollständige Konfiguration vom System übernommen. Beispiel:

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h1m with 0 errors on Tue Feb  2 13:54:30 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0  377M resilvered
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
```

Der Pool befindet sich jetzt wieder im Status ONLINE, und der ursprüngliche ausgefallene Datenträger (c1t0d0) wurde aus der Konfiguration entfernt.

Reparieren beschädigter Daten

In den folgenden Abschnitten wird beschrieben, wie Sie den Typ der Datenbeschädigung ermitteln und (falls möglich) Daten reparieren können.

- „Ermitteln der Art der Datenbeschädigung“ auf Seite 322
- „Reparatur beschädigter Dateien bzw. Verzeichnisse“ auf Seite 323
- „Reparieren von Schäden am gesamten ZFS-Speicher-Pool“ auf Seite 324

Zur Minimierung des Risikos von Datenbeschädigung nutzt ZFS Prüfsummenberechnung, Redundanz und Datenselbstheilung. Dennoch können Datenbeschädigungen auftreten, wenn ein Pool nicht redundant ist. Dies ist der Fall, wenn sich der Pool zum Zeitpunkt der Beschädigung in beeinträchtigtem Zustand befindet oder mehrere Datenkopien durch unvorhergesehene Ereignisse beschädigt werden. Unabhängig von der Ursache ist das Ergebnis dasselbe: Daten werden beschädigt und sind deswegen nicht mehr verfügbar. Die erforderlichen Abhilfemaßnahmen hängen von der Art der beschädigten Daten und ihrem relativen Wert ab. Es können zwei grundlegende Arten von Daten beschädigt werden:

- Pool-Metadaten – Damit Pools geöffnet werden können und auf Datasets zugegriffen werden kann, muss ZFS eine Reihe spezieller Daten verarbeiten. Wenn diese Daten beschädigt sind, stehen der gesamte Pool bzw. ein Teil der Datasets nicht mehr zur Verfügung.
- Objektdaten – In diesem Fall sind Daten innerhalb einer bestimmten Datei bzw. eines Verzeichnisses beschädigt. Dieses Problem kann dazu führen, dass auf einen Teil dieser Datei bzw. dieses Verzeichnisses nicht mehr zugegriffen werden kann oder das betreffende Objekt in seiner Gesamtheit nicht mehr verfügbar ist.

Daten werden während des Normalbetriebs und der Datenbereinigung auf Integrität überprüft. Weitere Informationen zum Überprüfen der Integrität von Pool-Daten finden Sie unter „Überprüfen der Integrität des ZFS-Dateisystems" auf Seite 301.

Ermitteln der Art der Datenbeschädigung

Der Befehl `zpool status` zeigt standardmäßig nur an, dass eine Datenbeschädigung aufgetreten ist, es ist jedoch nicht ersichtlich, wo sie sich ereignet hat. Beispiel:

```
# zpool status monkey
  pool: monkey
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: http://www.sun.com/msg/ZFS-8000-8A
 scrub: scrub completed after 0h0m with 8 errors on Tue Jul 13 13:17:32 2010
config:

        NAME        STATE     READ WRITE CKSUM
        monkey      ONLINE       8     0     0
          c1t1d0    ONLINE       2     0     0
          c2t5d0    ONLINE       6     0     0

errors: 8 data errors, use '-v' for a list
```

Ein Fehler zeigt lediglich an, dass zu einem bestimmten Zeitpunkt ein Fehler aufgetreten ist. Diese Fehler müssen nicht mehr notwendigerweise im System vorhanden sein. Unter normalen Bedingungen ist dies der Fall. Bestimmte zeitweilige Ausfälle können zu Datenbeschädigungen führen, die nach dem Ende des Ausfalls automatisch behoben werden. Bei einer vollständigen Bereinigung des Pools wird jeder aktive Datenblock im Pool untersucht, und das Fehlerprotokoll wird nach Abschluss der Bereinigung geleert. Wenn Sie sehen, dass die betreffenden Fehler im Pool nicht mehr auftreten und Sie nicht auf den Abschluss des Bereinigungsvorgangs warten möchten, können Sie alle Fehler im Pool mithilfe des Befehls `zpool online` löschen.

Wenn die Datenbeschädigung in Metadaten für den gesamten Pool aufgetreten ist, sieht die Befehlsausgabe etwas anders aus. Beispiel:

```
# zpool status -v morpheus
  pool: morpheus
    id: 1422736890544688191
 state: FAULTED
status: The pool metadata is corrupted.
action: The pool cannot be imported due to damaged devices or data.
   see: http://www.sun.com/msg/ZFS-8000-72
config:

        morpheus    FAULTED   corrupted data
          c1t10d0   ONLINE
```

Ist ein Pool beschädigt, wird er in den Status `FAULTED` versetzt, da er nicht die erforderliche Datenredundanz gewährleisten kann.

Reparatur beschädigter Dateien bzw. Verzeichnisse

Ist eine Datei oder ein Verzeichnis beschädigt, kann es je nach Art der Datenbeschädigung sein, dass das System noch immer funktioniert. Schäden sind praktisch nicht wieder rückgängig zu machen, wenn im System keine brauchbaren Datenkopien vorhanden sind. Wenn die betreffenden Daten wertvoll sind, müssen Sie diese aus Sicherungskopien wiederherstellen. Auch in diesem Fall kann es sein, dass Sie nach dieser Datenbeschädigung den Normalbetrieb wiederherstellen können, ohne den gesamten Pool wiederherstellen zu müssen.

Wenn sich der Schaden innerhalb eines Dateidatenblocks befindet, kann die betreffende Datei sicher gelöscht werden, wodurch der Fehler aus dem System entfernt wird. Mit dem Befehl `zpool status -v` können Sie eine Liste von Dateinamen mit dauerhaften Fehlern ausgeben. Beispiel:

```
# zpool status -v
  pool: monkey
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: http://www.sun.com/msg/ZFS-8000-8A
 scrub: scrub completed after 0h0m with 8 errors on Tue Jul 13 13:17:32 2010
config:

        NAME        STATE     READ WRITE CKSUM
        monkey      ONLINE       8     0     0
          c1t1d0    ONLINE       2     0     0
          c2t5d0    ONLINE       6     0     0

errors: Permanent errors have been detected in the following files:
```

```
/monkey/a.txt
/monkey/bananas/b.txt
/monkey/sub/dir/d.txt
monkey/ghost/e.txt
/monkey/ghost/boo/f.txt
```

Es folgen Erläuterung zur Liste von Dateinamen mit dauerhaften Fehlern:

- Wenn der vollständige Pfad zur Datei gefunden wurde und das Dataset eingehängt ist, wird der vollständige Pfad zur Datei angezeigt. Beispiel:

  ```
  /monkey/a.txt
  ```
- Wenn der vollständige Pfad zur Datei gefunden wurde, das Dataset jedoch nicht eingehängt ist, wird der Dataset-Name ohne vorangestellten Schrägstrich (/), gefolgt vom Pfad zur Datei innerhalb des Datasets angezeigt. Beispiel:

  ```
  monkey/ghost/e.txt
  ```
- Wenn die Objektnummer zu einem Dateipfad nicht erfolgreich konvertiert werden kann, weil ein Fehler auftrat oder dem Objekt kein realer Pfad zugewiesen ist (z. B. bei `dnode_t`), dann wird der Dataset-Name, gefolgt von der Objektnummer angezeigt. Beispiel:

  ```
  monkey/dnode:<0x0>
  ```
- Wenn ein Objekt im Metaobjekt-Set (MOS) beschädigt ist, wird ein spezielles Tag `<metadata>`, gefolgt von der Objektnummer, angezeigt.

Wenn Metadaten einer Datei bzw. eines Verzeichnisses beschädigt sind, kann die betreffende Datei nur an einen anderen Ort verschoben werden. Sie können Dateien bzw. Verzeichnisse sicher an eine unkritische Stelle kopieren, sodass das ursprüngliche Objekt am Ausgangsort wiederhergestellt werden kann.

Reparieren von Schäden am gesamten ZFS-Speicher-Pool

Wenn Pool-Metadaten beschädigt sind und der Pool dadurch nicht geöffnet oder importiert werden kann, stehen folgende Optionen zur Verfügung:

- Versuchen Sie, den Pool mithilfe des Befehls `zpool clear - F` oder `zpool import -F` wiederherzustellen. Mithilfe dieser Befehle wird versucht, die letzten Pool-Transaktionen zurückzusetzen, um den Pool wiederherzustellen. Sie können den Befehl `zpool status` verwenden, um einen beschädigten Pool und die empfohlenen Wiederherstellungsschritte anzuzeigen. Beispiel:

  ```
  # zpool status
    pool: tpool
   state: FAULTED
  status: The pool metadata is corrupted and the pool cannot be opened.
  action: Recovery is possible, but will result in some data loss.
          Returning the pool to its state as of Wed Jul 14 11:44:10 2010
  ```

```
        should correct the problem.  Approximately 5 seconds of data
        must be discarded, irreversibly.  Recovery can be attempted
        by executing 'zpool clear -F tpool'.  A scrub of the pool
        is strongly recommended after recovery.
   see: http://www.sun.com/msg/ZFS-8000-72
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
         tpool      FAULTED      0     0     1  corrupted data
           c1t1d0   ONLINE       0     0     2
           c1t3d0   ONLINE       0     0     4
```

Für die oben beschriebene Wiederherstellung ist der folgende Befehl zu verwenden:

```
# zpool clear -F tpool
```

Wenn Sie versuchen, einen beschädigten Pool zu importieren, wird eine Meldung wie die folgende angezeigt:

```
# zpool import tpool
cannot import 'tpool': I/O error
        Recovery is possible, but will result in some data loss.
        Returning the pool to its state as of Wed Jul 14 11:44:10 2010
        should correct the problem.  Approximately 5 seconds of data
        must be discarded, irreversibly.  Recovery can be attempted
        by executing 'zpool import -F tpool'.  A scrub of the pool
        is strongly recommended after recovery.
```

Für die oben beschriebene Wiederherstellung ist der folgende Befehl zu verwenden:

```
# zpool import -F tpool
Pool tpool returned to its state as of Wed Jul 14 11:44:10 2010.
Discarded approximately 5 seconds of transactions
```

Wenn der beschädigte Pool in der Datei `zpool.cache` enthalten ist, wird das Problem behoben, sobald das System neu gestartet wird, und die Beschädigung des Pools wird über den Befehl `zpool status` angezeigt. Wenn der Pool nicht in der Datei `zpool.cache` enthalten ist, kann er nicht importiert oder geöffnet werden, und es werden Meldungen angezeigt, die den beschädigten Pool betreffen, wenn Sie versuchen, den Pool zu öffnen.

- Wenn der Pool durch die oben beschriebene Wiederherstellung nicht wiederhergestellt werden kann, müssen Sie den Pool und alle seine Daten aus einer Sicherungskopie wiederherstellen. Das eingesetzte Verfahren hängt weitgehend von der Pool-Konfiguration und der Datensicherungsstrategie ab. Zuerst sollten Sie die mit dem Befehl `zpool status` angezeigte Konfiguration speichern, um diese nach dem Löschen des Pools wiederherstellen zu können. Löschen Sie den Pool dann mit dem Befehl `zpool destroy -f`. Legen Sie sich an sicherer Stelle eine Datei an, in der die Struktur der Datasets sowie die einzelnen lokal gesetzten Eigenschaften beschrieben sind, da diese Informationen nicht mehr zugänglich sind, wenn auf den Pool nicht mehr zugegriffen werden kann. Mithilfe der Pool-Konfiguration und der Dataset-Struktur können Sie nach dem Löschen des Pools die vollständige Konfiguration wiederherstellen. Danach können Sie mithilfe eines beliebigen Wiederherstellungsverfahrens den Pool wieder mit Daten „auffüllen“.

Reparieren eines Systems, das nicht hochgefahren werden kann

ZFS soll trotz möglicher Fehler robust und stabil sein. Dennoch können Softwarefehler oder bestimmte unerwartete Probleme zum Systemabsturz führen, wenn auf einen Pool zugegriffen wird. Jeder Pool muss im Rahmen des Systemstarts geöffnet werden, was bedeutet, dass Fehler beim Zugreifen auf einen Pool in eine Systemabsturz-Neustart-Schleife münden können. Als Ausweg aus einer solchen Situation muss ZFS mitgeteilt werden, Pools beim Systemstart zu ignorieren.

ZFS unterhält in `/etc/zfs/zpool.cache` einen internen Cache aktiver Pools und ihrer Konfiguration. Der Speicherort und der Inhalt dieser Datei sind nicht öffentlich zugänglich und Änderungen unterworfen. Wenn ein System nicht mehr hochgefahren werden kann, sollten Sie es mit der Meilenstein-Option `none` (`-m milestone=none`) booten. Nach dem Hochfahren des Systems hängen Sie das Root-Dateisystem ohne Schreibschutz ein. Anschließend benennen Sie die Datei `/etc/zfs/zpool.cache` um oder verschieben diese. Dadurch ?vergisst“ ZFS, dass im System Pools vorhanden sind und greift nicht auf den schadhaften Pool zu, der das Problem verursacht hat. Danach können Sie durch Absetzen des Befehls `svcadm milestone all` den normalen Systemzustand wiederherstellen. Beim Booten von einem alternativen Root-Verzeichnis zu Reparaturzwecken können Sie ein ähnliches Verfahren anwenden.

Wenn das System wieder läuft, können Sie versuchen, den Pool mithilfe des Befehls `zpool import` zu importieren. Dadurch tritt jedoch wahrscheinlich der gleiche Fehler wie beim Systemstart auf, da dieser Befehl zum Zugriff auf Pools das gleiche Verfahren verwendet. Wenn das System mehrere Pools besitzt, gehen Sie wie folgt vor:

- Benennen Sie die Datei `zpool.cache` um oder verschieben Sie sie in ein anderes Verzeichnis (siehe oben).
- Finden Sie heraus, bei welchem Pool Probleme auftreten, indem Sie sich mithilfe des Befehls `fmdump -eV` die Pools, für die kritische Fehler gemeldet wurden, anzeigen lassen.
- Importieren Sie die Pools nacheinander. Lassen Sie dabei die Pools aus, bei denen Probleme aufgetreten sind, die in der Ausgabe des Befehls `fmdump` angezeigt werden.

ANHANG A

Oracle Solaris ZFS-Versionsbeschreibungen

In diesem Anhang werden die verfügbaren ZFS-Versionen und deren Funktionen beschrieben, sowie das Solaris-Betriebssystem, das die jeweilige ZFS-Version und die Funktionen bereitstellt.

Dieser Anhang enthält folgende Abschnitte:

Überblick über ZFS-Versionen

Mit einer speziellen ZFS-Version, die in den Solaris-Versionen enthalten ist, werden neue ZFS-Pool- und Dateisystemfunktionen bereitgestellt. Mit dem Befehl `zpool upgrade` oder `zfs upgrade` kann festgestellt werden, ob die Version, zu der ein Pool oder Dateisystem gehört, niedriger als die gerade laufende Solaris-Version ist. Mit diesen Befehlen können Sie auch Ihre Pool- und Dateisystemversionen aktualisieren.

Informationen zu den Befehlen `zpool upgrade` und `zfs upgrade` finden Sie unter „Upgrade von ZFS-Dateisystemen (zfs upgrade)“ auf Seite 33 und „Aktualisieren von ZFS-Speicher-Pools“ auf Seite 122.

ZFS-Poolversionen

In der folgenden Tabelle sind die ZFS-Poolversionen aufgelistet, die in den Solaris-Versionen zur Verfügung stehen.

Version	Solaris 10	Beschreibung
1	Solaris 10 6/06	Ursprüngliche ZFS-Version
2	Solaris 10 11/06	Ditto blocks (replicated metadata)
3	Solaris 10 11/06	Hot spares and double parity RAID-Z
4	Solaris 10 8/07	zpool history
5	Solaris 10 10/08	`gzip`-Komprimierungsalgorithmus
6	Solaris 10 10/08	`Pool-Eigenschaft bootfs`
7	Solaris 10 10/08	Separate Intent-Log-Geräte
8	Solaris 10 10/08	Delegated Administration
9	Solaris 10 10/08	Eigenschaften `refquota` und `refreservation`
10	Solaris 10 5/09	Cache-Geräte
11	Solaris 10 10/09	Verbesserte Bereinigung
12	Solaris 10 10/09	Snapshot-Eigenschaften
13	Solaris 10 10/09	Eigenschaft `snapused`
14	Solaris 10 10/09	Eigenschaft `aclinherit` `passthrough-x`
15	Solaris 10 10/09	Speicherplatzberechnung für Benutzer und Gruppen
16	Solaris 10 9/10	Unterstützung der Eigenschaft `stmf`
17	Solaris 10 9/10	RAID-Z-Konfiguration mit dreifacher Parität
18	Solaris 10 9/10	Aufbewahrung von Snapshots
19	Solaris 10 9/10	Entfernen von Protokolliergeräten
20	Solaris 10 9/10	`Komprimierung mit ZLIe` (Zero-Length Encoding)
21	Solaris 10 9/10	Reserviert
22	Solaris 10 9/10	Empfangene Eigenschaften

ZFS-Dateisystemversionen

In der folgenden Tabelle sind die ZFS-Dateisystemversionen aufgelistet, die in den Solaris-Versionen zur Verfügung stehen.

Version	Solaris 10	Beschreibung
1	Solaris 10 6/06	Ursprüngliche ZFS-Dateisystemversionen

Version	Solaris 10	Beschreibung
2	Solaris 10 10/08	Erweiterte Verzeichniseinträge
3	Solaris 10 10/08	Eindeutiger Bezeichner für Unabhängigkeit von Groß-/Kleinschreibung und das Dateisystem (FUID)
4	Solaris 10 10/09	`userquota`- und `groupquota`-Eigenschaften

Index

B

C

D

E

H

I

J

K

L

M

N

S